Q-Chem User's Guide

Version 1.2
December 1998

Version 1.2

December 1998

The information in this document applies to version 1.2 of Q-Chem.

December 1998
User's Guide

Authored and Produced by:
Dr. Jeremy P. Dombroski

Edited by:
Dr. Thomas R. Furlani

Published by:
Q-Chem, Inc.
Four Triangle Lane
Suite 160
Export, PA 15632-9255

Customer Support:	**Telephone:**	(724) 325-9969
	Facsimile:	(724) 325-9560
	email:	support@q-chem.com
	website:	http://www.q-chem.com

PREFACE

Dear Q-Chem User,

As the 1990's draw to a close, we are witnessing unprecedented activity within the Q-Chem Project. I am pleased to report that the solid growth that we enjoyed during 1998 has continued into 1999 and I am especially happy to announce that our strengths have now been enhanced by the addition of Prof. John Pople (Northwestern University) to Q-Chem's Board of Directors. Prof. Pople's name is familiar to all quantum chemists and his receipt of the 1998 Nobel Prize for Chemistry was a fitting testimony to his numerous contributions to the subject over the past 50 years. We are thrilled by the confidence that he has placed in Q-Chem and we look forward enthusiastically to the scientific and corporate contributions that he will make within the leadership of the company. Prof. Pople joins Prof. Fritz Schaefer (University of Georgia) and Mr John Stott (Eastside Holdings Ltd), whom we were lucky enough to attract to the Board last year. It is very reassuring to be guiding Q-Chem forward in partnership with men of such stature and experience.

Our mission is to provide the fastest, most innovative and most useful Quantum Chemistry software package available and our pursuit of this goal has been refreshed and invigorated by this recent expansion in Q-Chem's intellectual leadership. When one combines this with the ongoing research efforts led by Jing Kong (Q-Chem, Inc.), Tom Furlani (SUNY, Buffalo), Peter Gill (Cambridge) and Martin Head-Gordon (UC, Berkeley), it becomes clear that Q-Chem is strongly positioned to lead the development of quantum chemistry software into the third millennium.

Q-Chem's status at the cutting edge of theoretical and computational development continues to be acknowledged by a variety of funding bodies. During 1998, we completed our NSF-funded KWIK project to devise a new approach to the rate-limiting step in DFT calculations. This year we will complete our USAF-funded Local Correlation project to develop techniques for rapid MP2 and CCSD calculations. We are about to embark upon an NSF-funded Gridless DFT project to develop analytically integrable density functionals and we also anticipate that we will continue our ongoing NIH-funded Parallel Q-Chem project to develop and implement code for shared- and distributed-memory machines. It is extremely exciting to watch projects like these evolve in a few years from back-of-the-envelope infancy to full-fledged maturity in a released version of the Q-Chem package.

To show our appreciation for your continued support and to celebrate Prof. Pople's decision to join Q-Chem, we are providing complimentary upgrades to Q-Chem 1.2 for all of our users. I should point out that the license for the Q-Chem 1.2 software is *identical to that for your existing Q-Chem 1.1 software and, therefore, if you are an existing Q-Chem customer you do not need to contact Q-Chem Inc. for a new license.*

We are grateful to a number of our customers and testers who told us about bugs in Q-Chem 1.1. We rely on such feedback to help us improve our package and Q-Chem 1.2 corrects a number of bugs that were in earlier versions. In addition, Q-Chem 1.2 includes the following new features:

- *Analytical second derivatives for CIS theory.*

  Users can now compute the vibrational frequencies of molecular excited states at the CIS level. We thank Dave Maurice for his hard work and ingenuity in developing this functionality.

- *Coulomb-attenuated (CASE) calculations.*

  Users can now investigate the effects of Coulomb attenuation on theoretical model chemistries. We thank Ross Adamson for interfacing his efficient CASE package to Q-Chem.

- *Geometry Optimization with General Constraints.*

  Users can now specify constraints in a number of ways that are completely general. We thank Jon Baker for assisting us with the interface between his OPTIMIZE package and Q-Chem.

We expect that you will find Q-Chem 1.2 a useful addition to your software collection and we hope that you will want to maintain your competitive edge by acquiring future versions of the code. Within the next few months, we plan to release versions of Q-Chem with analytical DFT frequencies, effective core potentials (pseudopotentials) and coupled cluster theory. We will also soon release the first parallel version of Q-Chem.

Have you considered joining the Q-Chem Membership Plan (QMP)? This is the successor to our popular Maintenance Scheme (QSMS) and was introduced at the beginning of this year. For an annual membership fee (40% of the original purchase price of the software), members of the scheme have priority access to Q-Chem's free email support service (support@q-chem.com) and automatically receive free upgrades to all future releases of their version of the Q-Chem software. For further information about membership, browse www.q-chem.com or write to info@q-chem.com

Yours sincerely,

Peter M.W. Gill
President

TABLE OF CONTENTS

CHAPTER 1 INTRODUCTION

1.1 ABOUT THIS MANUAL

This manual is intended as a general purpose user's guide for Q-Chem, a modern electronic structure program. The manual contains background information that describes Q-Chem methods and user-selected parameters. It is assumed that the user has some familiarity with the UNIX environment, an ASCII file editor and a basic understanding of quantum chemistry.

The manual is divided into nine chapters and four appendices which are briefly summarized below. After installing Q-Chem and making necessary adjustments to your user account, it is recommended that particular attention be given to Chapters Three and Four. The latter chapter has been formatted so that advanced users can quickly find the information they require, while supplying new users with a moderate level of important background information. This format has been maintained throughout the manual, and every attempt has been made to guide the user forward and backward to any relevant information in each section so that a logical progression through this manual, while recommended, is not necessary.

1.2 CHAPTER SUMMARIES

Chapter 1: General overview of the Q-Chem program, its features and capabilities, the people behind it and contact information
Chapter 2: Procedures to install, test and run Q-Chem on your machine
Chapter 3: Basic attributes of the Q-Chem command line input
Chapter 4: Running calculations using the many ground state methods available to Q-Chem users
Chapter 5: Running excited state calculations
Chapter 6: Using Q-Chem's built-in basis sets and running user-defined basis sets
Chapter 7: Options available for determining potential surface critical points such as transition states and local minima
Chapter 8: Techniques available for computing molecular properties and performing wavefunction analysis
Chapter 9: Important customization options available to enhance user flexibility
Appendix A: OPTIMIZE package used in Q-Chem for determining Molecular Geometry Critical Points
Appendix B: Q-Chem's AOINTS library, which contains some of the fastest two-electron integral codes currently available
Appendix C: *$rem* variable reference
Appendix D: Sample input files (a sample file is provided for most major Q-Chem functions; these files are also available in the release material)

1.3 CONTACT INFORMATION

1.3.1 GENERAL INQUIRIES AND SALES

For general information regarding broad aspects and features of the Q-Chem program, see Q-Chem's WWW home page (http://www.q-chem.com). Alternatively, contact Q-Chem, Inc. headquarters:

Q-Chem, Inc.
Four Triangle Lane
Suite 160
Export, PA 15632-9255
Telephone: (724) 325-9969
Fax: (724) 325-9560
email: sales@q-chem.com
support@q-chem.com
info@q-chem.com

1.3.2 CUSTOMER SUPPORT

Full customer support is promptly provided though telephone or email for those customers who have purchased Q-Chem's maintenance contract. The maintenance contract offers free customer support and discounts on future releases and updates. For details of the maintenance contract see Q-Chem's home page (http://www.q-chem.com).

1.4 Q-CHEM, INC.

Q-Chem, Inc. is based in Export, Pennsylvania and was founded in 1993. The board of directors includes leading quantum chemistry software developers - Martin Head-Gordon (Berkeley), Peter Gill (Cambridge), Fritz Schaefer (Georgia) and John Pople (Northwestern). Together with their collaborators, they created Q-Chem, the first commercially available quantum chemistry program capable of analyzing large structures in practical amounts of time.

In order to create this technology, the founders of Q-Chem, Inc. built entirely new methodologies from the ground up, using the latest algorithms and modern programming techniques. Since 1993, well over 20 man-years have been devoted to the development of the Q-Chem program.

Although the task of starting over is difficult, it enabled the developers to use the latest breakthroughs in Object Oriented Programming (OOP) to implement a revolutionary new program structure. With scientists worldwide expressing a need for more advanced capabilities to solve their ever more complex problems, Q-Chem, Inc. are confident that Q-Chem will become the leading quantum chemistry software program entering the new millennium. Q-Chem, Inc. is positioning itself to become the new leading quantum chemistry software developer.

1.5 COMPANY MISSION

The mission of Q-Chem, Inc. is to develop, distribute and support innovative quantum chemistry software for industrial, government and academic researchers in the chemical, petrochemical, biochemical, pharmaceutical and material sciences.

1.6 Q-CHEM FEATURES

Quantum chemistry methods have proven invaluable for studying chemical and physical properties of molecules. The Q-Chem system brings together a variety of advanced computational methods and tools in an integrated *ab initio* software package, greatly improving the speed and accuracy of calculations being performed. In addition, Q-Chem will accommodate far large molecular structures than previously possible and with no loss in accuracy, thereby bringing the power of quantum chemistry to critical research projects for which this tool was previously unavailable.

1.6.1 CAPABILITIES

- Most advanced SCF single point energy capability currently available
    - ◊ Linear scaling computation for all SCF theoretical methods
    - ◊ Ultra-fast integral computation
    - ◊ Efficient convergence acceleration methods
- Automated Geometry and Transition Structure Optimization
    - ◊ Optimizes in Cartesian, Z-matrix or delocalized internal coordinates
    - ◊ Eigenvector Following (EF) algorithm for minima and transition states
    - ◊ GDIIS algorithm for minima
    - ◊ Can impose bond angle, dihedral angle (torsion) or out-of-plane bend constraints
    - ◊ Freezes atoms in Cartesian coordinates
    - ◊ Desired constraints do not need to be satisfied in the starting structure
    - ◊ Geometry optimization in the presence of fixed point charges
- Vibrational Spectra
- Electronic Excitation Spectra
    - ◊ Excitation energies calculated at the CIS, RPA, XCIS and CIS(D) levels
    - ◊ Transition moments and oscillator strengths
    - ◊ Visualization *via* attachment-detachment analysis at the CIS level of theory
- Electrostatic potentials
- Molecular Orbital and Density Plotting
- Seamless integration with supported GUI plotting facilities
- Natural Bond Orbital Analysis *via* NBO

1.6.2 Theoretical Methods

- Continuous Fast Multipole Method (CFMM)
    - ◊ Linear-cost calculation of Coulomb interactions
    - ◊ Finds *exact* Coulomb energy; no approximations are made
- QCTC
- Hartree-Fock Theory
    - ◊ Automated optimal blend of in-core and direct SCF methods
    - ◊ Fast exchange algorithms for large molecules (ONX and LinK)
- Local and Gradient-Corrected DFT functionals
    - ◊ Slater, Becke, GGA91 and Gill '96 exchange functionals
    - ◊ VWN, PZ81, Wigner, Perdew86, LYP and GGA91 correlation functionals
    - ◊ EDF1 exchange-correlation functional
    - ◊ Linear cost XC algorithm (CPU and memory) for large molecules
- Hybrid DFT Methods
    - ◊ B3LYP
    - ◊ B3P
    - ◊ User-definable hybrids
- MP2 Perturbation Theory
    - ◊ Energy *via* direct and semi-direct methods
    - ◊ Analytical gradient *via* efficient semi-direct methods
    - ◊ Proper treatment of frozen orbitals in analytical gradients
    - ◊ Excited states treated *via* the CIS(D) method
- CIS, RPA, XCIS, CIS(D) Methods for Excited States
    - ◊ Restricted, unrestricted and restricted open-shell CIS supported
    - ◊ Energies, gradients and second derivatives available for restricted and unrestricted CIS with energies available for all other methods
    - ◊ CIS implementations are direct methods designed for large molecules
    - ◊ CIS uses full non-Abelian symmetry
    - ◊ CIS(D) treats electron correlation effects on electronic transitions
    - ◊ Efficient direct and semi-direct implementation of CIS(D) makes the cost of this method (per state) similar to MP2
    - ◊ Can restrict calculation to user-specified states only
- AOINTS package for Two-Electron Integrals
    - ◊ Incorporates the latest advances in high performance integrals technology
    - ◊ COLD PRISM
    - ◊ J Matrix engine

1.7 Highlighted Features

Developed by Q-Chem, Inc. and its collaborators, fundamental features include COLD PRISM, CFMM, CIS(D), QCTC/ONX and OPTIMIZE packages. The features, which are highlighted below, are elaborated in later relevant sections.

1.7.1 Theoretical Advancements

COLD PRISM

The COLD PRISM is the latest in a number of high performance two-electron integral algorithms developed by Peter Gill and his collaborators at Massey University and the University of Cambridge. The development of COLD PRISM began with the realization that all methods for computing two-electron integral matrix elements involve four steps (represented by the COLD acronym), namely - contraction (C), operator (O), momentum (L) and density (D). This has culminated in the unification and augmentation of the previous PRISM and J engine methodologies into a generalised scheme, for the construction of two-electron matrix elements from shell-pair data.

Continuous Fast Multipole Method (CFMM)

One of the main driving forces in the evolution of Q-Chem is the implementation of the Continuous Fast Multipole Method (CFMM) developed by Chris White at the University of California at Berkeley. This enables Q-Chem to calculate the electronic Coulomb interactions (the rate-limiting step in large DFT calculations) in less time than other programs, and the time saved actually increases as the molecule becomes larger.

CIS(D)

The CIS(D) method developed by David Maurice, Manabu Oumi and Martin Head-Gordon at the University of California at Berkeley has been implemented for a correlated treatment of electronic transitions. With an efficient direct and semi-direct implementation of CIS(D), the computational cost of this method per excited state is similar to that of an MP2 calculation. CIS(D) is a useful method for *ab initio* calculations of electronic excitations in large radicals.

QCTC/ONX

Conventional HF and Hybrid HF/DFT calculations are limited by calculation of both Coulomb and exact exchange matrices. Q-Chem includes the quantum chemical tree code (QCTC) for linear scaling calculation of the Coulomb matrix and ONX for rapid computation of the exact exchange matrix. The algorithms, developed by Matt Challacombe and Eric Schwegler at the University of Minnesota, are numerically equivalent to conventional integral-based algorithms, but scale linearly for complicated insulating systems like water clusters and protein molecules.

OPTIMIZE

The Q-Chem program incorporates the latest version of Jon Baker's OPTIMIZE package, containing a suite of state-of-the-art algorithms for geometry optimization including the extremely efficient delocalized internal coordinates. Dr. Baker wrote the optimization algorithms in the Spartan package and the optimization code in the Biosym-distributed versions of DMol, Turbomole and Zindo.

1.7.2 Graphical User Interface (GUI)

HyperChem

HyperChem special Release 4.5 from HyperCube, Inc. defines a GUI for Q-Chem using the new Chemist Development Kit (CDK). Powerful modelling features allow for the rapid creation of molecular structures to be used as input data. This GUI allows the user to prepare Q-Chem input files quickly and efficiently without resorting to low level input file manipulation. HyperChem can monitor Q-Chem's computational analysis interactively and can graphically interpret Q-Chem's results including molecular orbitals, electron densities, UV-vis spectra and normal modes of vibration.

- Molecular orbitals
- Electron densities
- Vibrational spectra
- Electronic spectra

Unichem

Unichem™ from Oxford Molecular Group provides users with a single point of access to the entire simulation process. This product includes a molecular builder, an easy-to-use job setup and launch capability, an interactive job monitor and comprehensive analysis capabilities including the ability to display such results as:

- Molecular orbitals
- Charge and spin densities
- Transition densities
- Vibrational spectra

1.8 Current Development and Future Releases

All details of functionality currently under development, information relating to future releases, and patch information are regularly updated on the Q-Chem web page (http://www.q-chem.com). Users are referred to this page for updates on developments, release information and further information on ordering and licenses. For any additional information, please contact Q-Chem, Inc. headquarters.

1.9 Citing Q-Chem

The official Q-Chem citation for this release is:

```
C.A. White, J. Kong, D.R. Maurice, T.R. Adams, J. Baker,
M. Challacombe, E. Schwegler, J.P. Dombroski, C. Ochsenfeld,
M. Oumi, T.R. Furlani, J. Florian, R.D. Adamson, N. Nair, A.M. Lee,
N. Ishikawa, R.L. Graham, A. Warshel, B.G. Johnson,
P.M.W. Gill and M. Head-Gordon, Q-Chem, Version 1.2,
Q-Chem, Inc., Pittsburgh, PA (1998).
```

CHAPTER 2 INSTALLATION

2.1 Q-CHEM INSTALLATION REQUIREMENTS

2.1.1 SOFTWARE

Q-Chem is provided already compiled and ready to run on your system.

The software required to run Q-Chem on your platform is minimal and includes:

- a suitable operating system
- FORTRAN, C++, BLAS, LAPACK runtime libraries (usually provided with your operating system)
- the Q-Chem installation media

2.1.2 OPERATING SYSTEM

Q-Chem has been optimized to run under the following operating systems:

- Cray UNICOS 8 & 9
- DEC OSF/1 v3.2 & v 4.0
- IBM AIX 4.1-3
- SGI IRIX 6.2-5
- Solaris 2.6
- UXP/V V10L20 X97121
- Linux

If you are unsure if Q-Chem will operate with your current operating system, or if you have another version of any of the above, consult the Q-Chem home page for an update of available operating systems, or contact Q-Chem’s customer support. It maybe possible to provide a version to match your specific configuration.

2.1.3 INSTALLATION MEDIA

Q-Chem is shipped on CD-ROM.

2.1.4 Hardware

Platforms

Q-Chem has been optimized for a number of computer platforms including:

- C90 and T3E Cray Supercomputers
- O2, Indigo^2, Octane, Origin, Onyx and Challenge Series SGI workstations
- IBM RS/6000 workstations
- IBM SP2
- DEC Alpha workstations
- DEC 8400
- UltraSPARC SUN workstations
- Fujitsu VPP
- Intel
- Others may be available on request

Consult the Q-Chem home page for updates of specific platform availability or contact Q-Chem's customer support service.

Memory

Q-Chem, Inc. has endeavoured to minimise memory requirements and maximise the efficiency of its use. A workable minimum amount of memory is about 16 MB, but upwards of 64 MB dramatically increases performance and the scope of application. Q-Chem also offers the ability for user control of important memory intensive aspects of the program, an important consideration for non-batch constrained multi-user systems.

Disk

The Q-Chem executables, shell scripts, samples and release notes require approximately 80-100 MB of disk space, depending on the platform. In order to maximise the capabilities of your copy of Q-Chem, additional disk space is required for scratch files created during execution; these are normally automatically deleted on termination of a job. The amount of disk space required for scratch files depends on the type of job, the size of the molecule and the basis set chosen. Around 100 MB would suffice for low-level single-user systems. Ideally, multiple-user systems and large molecule calculations require dedicated scratch filesystems. The default Q-Chem output, which is printed to the designated output file, is usually only a few KB's. This will be exceeded, of course, in difficult geometry optimizations, and in cases where users invoke non-default print options.

2.2 Installing Q-Chem

Users are referred to the installation guide supplied with the installation media for installation instructions pertinent to the release and platform. Should any difficulties arise during installation, please refer to the Q-Chem website (http://www.q-chem.com, FAQ's, telephone and facsimile numbers) or directly contact Q-Chem customer support (email: support@q-chem.com) for assistance.

2.3 LICENSE REQUIREMENTS

In order to run Q-Chem you must obtain the necessary encrypted license password file. The license consists of two files. Place these files, filenames *qchem.license.dat* and *qchem.aux*, in the *$QCAUX/license* directory.

Do not alter these files unless directed by Q-Chem, Inc.

2.4 ENVIRONMENT VARIABLES

Q-Chem requires four shell environment variables in order to run calculations.

QC defines the location of the Q-Chem directory structure. The *qchem.install* shell script determines this automatically.

QCAUX defines the location of the auxiliary information required by Q-Chem, which includes the license required to run Q-Chem. This defaults to *$QC/aux*. The user may redefine this location.

QCSCRATCH defines the directory in which all scratch files will be placed during a run. Note that many of the files become quite large for any given run, and it should be ensured that sufficient disk space is available. Abnormally terminated jobs, and jobs invoked with three arguments (saves key files between runs), may require extra disk space. The *QCSCRATCH* directory should be periodically checked for scratch files remaining from abnormally terminated jobs. *QCSCRATCH* defaults to the working directory.

QCPLATFORM defines the nature of the platform on which Q-Chem will be run and is used to optimize operating performance. This is determined by the install shell script, *$QC/qchem.install*. The currently supported options are:

Platform Identifier	Platform	Operating System
CRAY_C90	Cray Supercomputer	UNICOS 8 & 9
CRAY_T3E	Cray Supercomputer	UNICOS 8 & 9
DEC_ALPHA	Digital	OSF/1 v3.2 & 4.x
DEC_PMAX	Digital	OSF/1 v3.2 & 4.x
FUJITSU_VP	Fujitsu Supercomputer	UXP/V
IBM_SP2	IBM SP2	AIX 4.x
IBM_RS6K	IBM RS/6000	AIX 4.x
SGI_IRIX64	Silicon Graphics	IRIX 6.2-5
SUN_SOLARIS	SUN Sparc	Solaris 2.6
LINUX_Ix86	Intel	Linux (with glibc)

Table 2.1 *$QCPLATFORM* variables relating to specific platforms and operating systems.

2.5 USER ACCOUNT ADJUSTMENTS

In order for individual users to run Q-Chem, their user environment must be modified as follows:

- User file access permissions must be set so that the user can read, write and execute the necessary Q-Chem files. It may be advantageous to create a Q-Chem User's UNIX group on your machine and recursively change the group ownership of the Q-Chem files to that of the new group.
- A few lines need to be added to user login files or to the system default login files. The Q-Chem environment variables need to be defined and the Q-Chem set up file needs to be initiated prior to use of Q-Chem (once, on login).

2.5.1 EXAMPLE LOGIN FILE MODIFICATIONS

For users using the csh shell (or equivalent), add the following lines to their home directory *.cshrc* file:

```
#*****  Q-Chem Configuration Begin *****
setenv QC directory_name
setenv QCAUX directory_name
setenv QCSCRATCH directory_name
setenv QCPLATFORM platform_identifier
if (-e ${QC}/bin/qchem.setup) source ${QC}/bin/qchem.setup
unset noclobber
#*****  Q-Chem Configuration End *****
```

For users using the Bourne shell (or equivalent), add the following lines to their home directory *.profile* file:

```
#*****  Q-Chem Configuration Begin *****
QC=directory_name; export QC
QCAUX=directory_name; export QCAUX
QCSCRATCH=directory_name; export QCSCRATCH
QCPLATFORM=platform_identifier; export QCPLATFORM
noclobber=“”
if [ -e ${QC}/bin/qchem.setup.sh ] ; then
   . ${QC}/bin/qchem.setup.sh
fi
#*****  Q-Chem Configuration End *****
```

Alternatively, these lines can be added to system wide *profile* or *cshrc* files or their equivalents.

2.6 THE QCHEM.SETUP FILE

When sourced on login from the *.cshrc* (or *.profile*, or equivalent), the qchem.setup(.sh) file makes a number of changes to the operating environment to enable the user to fully exploit Q-Chem capabilities, without adversely affecting any other aspect of the login session. The file:

- defines a number of environment variables used by various parts of the Q-Chem program
- sets the default directory for *QCAUX*, if not already defined
- adjusts the *PATH* environment variable so that the user can access Q-Chem’s executables from the users working directory
- checks that the Q-Chem variable *QCPLATFORM* has been appropriately set.

2.7 RUNNING Q-CHEM

Once installation is complete and any necessary adjustments are made to the user account, the user is now able to run Q-Chem. There are two ways to invoke Q-Chem:

- *qchem* command line shell script
- supported Graphical User Interface

Using the Q-Chem command line shell script, *qchem*, is straightforward provided Q-Chem has been correctly installed on your machine and the necessary environment variables have been set in *.cshrc* or *.profile* (or equivalent) login files. If done correctly, necessary changes will have been made to the *PATH* variable automatically on login so that Q-Chem can be invoked from your working directory. The *qchem* shell script can be used in either of the following ways:

```
qchem infile outfile

qchem infile outfile save
```

where *infile* is the name of a suitably formatted Q-Chem input file (detailed in Chapter 3), and the *outfile* is the name of the file to which Q-Chem will place the job output information.

Note: If the *outfile* already exists in the working directory, it will be overwritten.

The use of the *save* command line variable allows the saving of key files between runs and is necessary when instructing Q-Chem to read information from previous jobs. By default, Q-Chem deletes its intermediate files at the end of a run.

The name of the input parameters *infile*, *outfile* and *save* can be chosen at the discretion of the user (usual UNIX file and directory name restrictions apply). It maybe helpful to use the same jobname for *infile* and *outfile*, but with varying suffixes. For example:

```
localhost-1> qchem water.input water.output &
```

invokes Q-Chem where the input is taken from *water.input* and the output is placed into *water.output*. The *&* places the job into the background so that you may continue to work in the current shell.

```
localhost-2> qchem water.com water.log water &
```

invokes Q-Chem where the input is assumed to reside in *water.com*, the output is placed into *water.log* and scratch files are saved in a directory *$QCSCRATCH/water/*.

For information regarding running Q-Chem using the supported GUIs, refer to the guide supplied with the GUI.

2.8 TESTING Q-CHEM

Q-Chem is shipped with a small number of test jobs, which are situated in the *$QC/samples* directory. If you wish to test your version of Q-Chem, run the test jobs in the samples directory and compare the output files with the reference files (suffixed *.ref*) of the same name.

These test jobs are not an exhaustive quality control test (a small subset of the test suite used at Q-Chem, Inc.), but they should all run correctly on your platform. However, if any fault is identified in these or any output files created by your version, do not hesitate to contact customer service immediately.

CHAPTER 3 Q-CHEM INPUTS

3.1 GENERAL FORM

A graphical interface is the simplest way to control Q-Chem. However, the low level command line interface is available to enable maximum customization and user exploitation of all Q-Chem features. The command line interface requires a Q-Chem input file which is simply an ASCII text file. This input file can be created using your favourite editor (*e.g.* vi, emacs, jot, *etc.*) following the basic steps outlined in the next few chapters.

Q-Chem's input mechanism uses a series of **keywords** to signal user input sections of the input file. As required, the Q-Chem program searches the input file for supported keywords. When Q-Chem finds a keyword, it then reads the section of the input file beginning at the keyword until that keyword section is terminated *$end*. A short description of all Q-Chem keywords is provided in Figure 3.1. The user **must** understand the function and format of the *$molecule* (section 3.2) and *$rem* (section 3.5) keywords, as these keyword sections are where the user places the molecular geometric information and job specification details.

The keywords *$rem* and *$molecule* are requisites of Q-Chem input files.

As each keyword has a different function, the format required for specific keywords varies somewhat, to account for the different specialised information (format requirements are summarised at the end of this chapter). But, because each keyword in the input file is sought out independently by the program as the information is required, the overall format requirements of Q-Chem input files are much less stringent. *e.g.*, it is not necessary to enter a user-defined basis set in a particular part of the input, if it is contained within the appropriate keyword (*$basis*) section and in the correct format.

`$molecule`	Contains the molecular coordinate input (input file requisite)
`$end`	Terminates each keyword section
`$rem`	Job specification and customization parameters (input file requisite)
`$basis`	Basis set information for user-defined basis sets (see Chapter 6)
`$comment`	User comments for inclusion into output file
`$external_charges`	External charges and their positions
`$multipole_field`	Details of a multipole field to apply
`$nbo`	Natural Bond Orbital package
`$occupied`	Guess orbitals to be occupied
`$opt`	Constraint definitions for geometry optimizations
`$xc_functional`	Details of user-defined DFT exchange-correlation functionals

Figure 3.1 Q-Chem user input section keywords

Notes: (1) Users are able to enter keyword sections in any order.
(2) Each keyword section must be terminated with the *$end* keyword.
(3) It is not necessary to have all keywords in an input file.
(4) Each keyword section will be described below.
(5) The entire Q-Chem input is case-insensitive.

In general, users will need to enter variables for the *$molecule* and *$rem* keyword section and are encouraged to add a *$comment* for future reference. The necessity of other keyword input will become apparent throughout the manual, and is summarised at the end of this Chapter.

See the Appendix and/or the *$QC/samples* directory with your release for specific examples of Q-Chem input using the keywords in Figure 3.1.

The second general aspect of Q-Chem input, is that there are effectively four input sources:

1. User input file (required)
2. *.qchemrc* file in *$HOME* (optional)
3. *preferences* files in *$QC/config* (optional)
4. Internal program defaults and calculation results (built-in)

These are summarised in order of preference in Figure 3.2. Thus, the input mechanism offers a program default over-ride for *all* users, default override for *individual* users and, of course, the input file provided by the user overrides all defaults. Refer to Chapter 9 for details of *.qchemrc* and *preferences*.

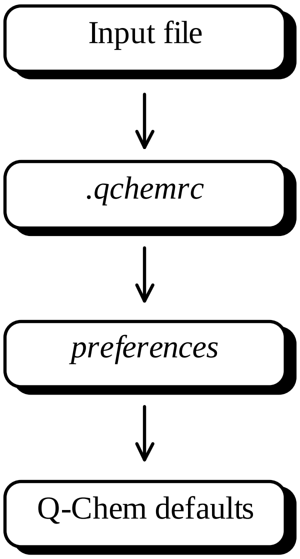

Figure 3.2 Diagram of input initialisation override settings. The higher mechanism indicates override preference of lower mechanisms.

Currently, Q-Chem only supports the *$rem* keyword in *.qchemrc* and *preferences* files.

3.2 MOLECULAR COORDINATE INPUT ($MOLECULE)

The *$molecule* section communicates to the program the charge, spin multiplicity and geometry of the molecule under investigation. The molecular coordinate input begins with two integers: the net charge and the spin multiplicity of the molecule. The net charge must be between -50 and 50, inclusively (0 for neutral molecules, 1 for cations, -1 for anions, *etc*.). The multiplicity must be between 1 and 10, inclusively (1 for a singlet, 2 for a doublet, 3 for a triplet, *etc*.). Each subsequent line of the molecular coordinate input corresponds to a single atom in the molecule (or dummy atom), irrespective of whether using *Z*-matrix internal coordinates or Cartesian coordinates.

Note: The coordinate system used for declaring an initial molecular geometry by default does not affect that used in a geometry optimization procedure. See the appendix which discusses the OPTIMIZE package in further detail.

Q-Chem begins all calculations by rotating and translating the user-defined molecular geometry into a Standard Nuclear Orientation whereby the centre of nuclear charge is placed at the origin. This is a standard feature of most quantum chemistry programs.

Note: Q-Chem ignores commas and equal signs, and requires all distances, positions and angles to be entered as Angstroms and degrees.

```
$molecule
0 1
O
H1 O distance
H2 O r H1 theta

distance = 1.0
theta = 104.5
$end
```

Figure 3.3 Example of molecular coordinate input for a water molecule in *Z*-matrix coordinates. Note that the *$molecule* input begins with the charge and multiplicity.

3.2.1 READING MOLECULAR COORDINATES FROM A PREVIOUS CALCULATION

Often users wish to perform several calculations in quick succession, whereby the later calculations rely on results obtained from previous calculations. For example, geometry optimization at a low level of theory, followed by vibrational analysis and then, perhaps, single point energy at a higher level. Rather than having the user manually transfer the coordinates from the output of the optimization to the input file of a vibrational analysis or single point energy calculation, Q-Chem can transfer them directly from job to job.

To achieve this requires that:

1. The *READ* variable is entered into the molecular coordinate input
2. Scratch files from a previous calculation have been saved. These may be obtained *explicitly* by using the save option across multiple job runs as described in chapter 2, or *implicitly* when running multiple calculations in one input file, as described later in this chapter.

```
$molecule
READ
$end
```

Figure 3.4 Reading a geometry from a prior calculation

3.2.2 Example

```
localhost-1> qchem job1.in job1.out job1
localhost-2> qchem job2.in job2.out job1
```

Example 3.1 In this example, the *job1* scratch files are saved in a directory *$QCSCRATCH/job1* and are then made available to the *job2* calculation.

Note: The program must be instructed to read specific scratch files by the input of *job2*.

Users are also able to use the *READ* function for molecular coordinate input using Q-Chem's batch job file (see later in this chapter).

3.2.3 Reading molecular Coordinates from another file

Users are able to use the *READ* function to read molecular coordinates from a second input file. The format for the coordinates in the second file follows that for standard Q-Chem input, and must be deliminated with the *$molecule* and *$end* keywords.

```
$molecule
READ      filename
$end
```

Figure 3.5 Reading molecular coordinates from another file. *filename* maybe given either as the full file path or, path relative to the working directory.

3.3 Cartesian Coordinates

Q-Chem can accept a list of *N* atoms and their *3N* Cartesian coordinates. The atoms can be entered either as atomic numbers or atomic symbols where each line corresponds to a single atom. The Q-Chem format for declaring a molecular geometry using Cartesian coordinates (in Angstroms) is:

```
atom      x-coordinate   y-coordinate   z-coordinate
```

3.3.1 Examples:

```
$molecule
0 1
8          0.000000        0.000000       -0.212195
1          1.370265        0.000000        0.848778
1         -1.370265        0.000000        0.848778
$end
```

Example 3.2 Atomic number Cartesian coordinate input for H_2O.

```
$molecule
0 1
O          0.000000        0.000000       -0.212195
H          1.370265        0.000000        0.848778
H         -1.370265        0.000000        0.848778
$end
```

Example 3.3 Atomic symbol Cartesian coordinate input for H_2O.

Notes:

- Atoms can be declared by either atomic number or symbol
- Coordinates can be entered either as variables/parameters or real numbers
    - Variables/parameters can be declared in any order
    - A single blank line separates parameters from the atom declaration

Once all the molecular Cartesian coordinates have been entered, terminate the Molecular Coordinate Input with the *$end* keyword.

3.4 Z-MATRIX COORDINATES

Z-matrix notation is one of the most common molecular coordinate input forms. The *Z*-matrix defines the positions of atoms relative to previously defined atoms using a length, an angle and a dihedral angle. Again, note that all bond lengths and angles must be in Angstroms and degrees.

Note: As with the Cartesian coordinate input method, Q-Chem begins a calculation by taking the user-defined coordinates and translating and rotating them into a Standard Nuclear Orientation.

The first three atom entries of a *Z*-matrix are different from the subsequent entries. The first *Z*-matrix line declares a single atom. The second line of the *Z*-matrix input declares a second atom, refers to the first atom and gives the distance between them. The third line declares the third atom, refers to either the first or second atom, gives the distance between them, refers to the remaining atom and gives the angle between them. All subsequent entries begin with an atom declaration, a reference atom and a distance, a second reference atom and an angle, a third reference atom and a dihedral angle. This can be summarised as:

1. First atom
2. Second atom, reference atom, distance
3. Third atom, reference atom A, distance between A and the third atom, reference atom B, angle defined by atoms A, B and the third atom
4. Fourth atom, reference atom A, distance, reference atom B, angle, reference atom C, dihedral angle (A, B, C and the fourth atom)
5. All subsequent atoms follow the same basic form as (4)

```
O1
O2 O1 OO
H1 O1 HO O2 HOO
H2 O2 HO O1 HOO H1 HOOH
```

Example 3.4 *Z*-matrix for hydrogen peroxide

Line 1 declares an oxygen atom (`O1`). Line 2 declares the second oxygen atom (`O2`), followed by a reference to the first atom (`O1`) and a distance between them denoted `OO`. Line 3 declares the first hydrogen atom (`H1`), indicates it is separated from the first oxygen atom (`O1`) by a distance `HO` and makes an angle with the second oxygen atom (`O2`) of `HOO`. Line 4 declares the fourth atom and the second hydrogen atom (`H2`), indicates it is separated from the second oxygen atom (`O2`) by a distance `HO` and makes an angle with the first oxygen atom (`O1`) of `HOO` and makes a dihedral angle with the first hydrogen atom (`H1`) of `HOOH`.

Some further points to note are:

- atoms can be declared by either atomic number or symbol
    - if declared by atomic number, connectivity needs to be indicated by *Z*-matrix line number
    - if declared by atomic symbol either
        - number similar atoms (*e.g.*, H1, H2, O1, O2 *etc.*) and refer connectivity using this symbol, or
        - indicate connectivity by the line number of the referred atom
- bond lengths and angles can be entered either as variables/parameters or real numbers
    - variables/parameters can be declared in any order
    - a single blank line separates parameters from the *Z*-matrix

All the following examples are equivalent in the information forwarded to the Q-Chem program. Example 3.5. type format is recommended :

```
$molecule
0 1
O1
O2 O1 OO
H1 O1 HO O2 HOO
H2 O2 HO O1 HOO H1 HOOH

OO=1.5
HOO =120.0
OH = 1.0
HOOH= 180.0
$end
```

Example 3.5 Using parameters to define bond lengths and angles, and using numbered symbols to define atoms and indicate connectivity.

```
$molecule
0 1
O1
O2 O1 1.5
H1 O1 1.0 O2 120.0
H2 O2 1.0 O1 120.0 H1 180.0
$end
```

Example 3.6 Not using parameters to define bond lengths and angles, and using numbered symbols to define atoms and indicate connectivity.

```
$molecule
0 1
8
8 1 OO
1 1 HO 2 HOO
1 2 HO 1 HOO 3 HOOH

OO=1.5
HOO=120.0
OH=1.0
HOOH=180.0
$end
```

Example 3.7 Using parameters to define bond lengths and angles, and referring to atom connectivities by line number.

```
$molecule
0 1
8
8 1 1.5
1 1 1.0 2 120.0
1 2 1.0 1 120.0 3 180.0
$end
```

Example 3.8 Referring to atom connectivities by line number, and entering bond length and angles directly.

Obviously, a number of the formats outlined above are less appealing to the eye and more difficult for us to interpret than the others, but each communicates *exactly* the same *Z*-matrix to the Q-Chem program.

3.4.1 DUMMY ATOMS

Dummy atoms are indicated by the identifier *X* and followed, if necessary, by an integer. (*e.g.*, *X1*, *X2*). Dummy atoms are often useful for molecules where symmetry axes and planes are not centred on a real atom, and have also been useful in the past for choosing variables for structure optimization and introducing symmetry constraints.

Note: Dummy atoms play no role in the quantum mechanical calculation, and are used merely for convenience in specifying other atomic positions or geometric variables.

3.5 JOB SPECIFICATION: THE $REM ARRAY CONCEPT

The *$rem* array is the means by which users convey to Q-Chem the type of calculation they wish to perform (level of theory, basis set, convergence criteria, *etc*.) The keyword *$rem* signals the beginning of the overall job specification. Within the *$rem* section the user inserts ***$rem* variables** (one per line) which define the essential details of the calculation. The format for entering *$rem* variables within the *$rem* keyword section of the input is:

```
REM_VARIABLE   OPTION    [Comment]
```

Figure 3.6 Format for declaring *$rem* variables in the *$rem* keyword section of the Q-Chem input file. Note, Q-Chem only reads the first two arguments on each line of *$rem*. All other text is ignored and can be used for placing short user comments.

The *$rem* array stores all details required to perform the calculation, and details of output requirements. It provides the flexibility to customize a calculation to specific user requirements. If a default *$rem* variable setting is indicated in this manual, the user does not have to declare the variable in order for the default to be initiated (*e.g*., the default *JOBTYPE* is a single point energy (*SP*)). Thus, to perform a single point energy calculation, the user does **not** need to set the *$rem* variable *JOBTYPE* to *SP*. However, to perform an optimisation, for example, it is necessary to override the program default by setting *JOBTYPE* to *OPT*.

A number of the *$rem* variables have been set aside for internal program use, as they represent variables automatically determined by Q-Chem (*e.g.*, the number of atoms, the number of basis functions). These need not concern the user.

User communication to the internal program *$rem* array comes in two general forms: (1) long term, machine-specific customization *via* the *.qchemrc* and *preferences* files (Chapter 9) and, (2) the Q-Chem input deck. There are many defaults already set within the Q-Chem program many of which can be overridden by the user. Checks are made to ensure that the user specifications are permissible (*e.g.*, integral accuracy is confined to 10^{-12}) and adjusted, if necessary. If adjustment is not possible, an error message is returned. Details of these checks and defaults will be given as they arise.

The user need not know all elements, options and details of the *$rem* array in order to fully exploit the Q-Chem program. Many of the necessary elements and options are determined automatically by the program, or the optimized default parameters, supplied according to the user's basic requirements, available disk and memory, and the operating system and platform.

3.6 $REM ARRAY FORMAT IN Q-CHEM INPUT

All data between the *$rem* keyword and the next appearance of *$end*, is assumed to be user *$rem* array input. On a single line for each *$rem* variable, the user declares the *$rem* variable, followed by a blank space (tab stop inclusive) and then the *$rem* variable option. It is recommended that a comment be placed following a space after the *$rem* variable option. *$rem* variables are case insensitive and a full listing is supplied in the appendix. Depending on the particular *$rem* variable, *$rem* options are entered either as a case-insensitive keyword, an integer value or logical identifier (true/false). The format for describing each *$rem* variable in this manual is as follows:

REM_VARIABLE

Gives a short description of what the variable controls

VARIABLE:

Defines the variable as either INTEGER, LOGICAL or STRING

DEFAULT:

Describes Q-Chem's internal default, if any exist

OPTIONS:

Lists options available for the user

RECOMMENDATION:

Gives a quick recommendation

The end of the *$rem* location declaration is signalled by the string *$end*.

```
$rem
rem_variable        option    [user_comment]
rem_variable        option    [user_comment]
...
...
$end
```

Figure 3.7 General format of the *$rem* section of the text input file.

Notes: (1) Erroneous lines will terminate the calculation
(2) Tab stops can be used to format input
(3) Entire lines can be commented by prefixing the line with an exclamation mark "!"

3.7 Minimum $rem Array Requirements

Although Q-Chem provides defaults for most *$rem* variables, the user will always have to stipulate a few others. For example, in a single point energy calculation, the minimum requirements will be *BASIS* (defining the basis set), *EXCHANGE* (defining the level of theory to treat exchange) and *CORRELATION* (defining the level of theory to treat correlation, if required).

```
$rem
BASIS               6-31G*    Just a small basis set
EXCHANGE            HF        Exact exchange
CORRELATION         MP2       MP2 energy
$end
```

Example 3.9 Example of minimum *$rem* requirements to run an MP2/6-31G* energy calculation.

3.8 Comments ($comment)

Users are able to add comments to the input file outside keyword input sections, which will be **ignored** by the program. This can be useful as reminders to the user, or perhaps, when teaching another user to set up inputs. Q-Chem has also provided a means of adding comments *via* the *$comment* which will be placed **into** the output file. For example, an initial geometry obtained at another level of theory, or from a publication, may be used to calculate other properties using Q-Chem. The source of the initial geometry can then automatically be placed into the output file as a comment.

Note: Currently the entire input deck is copied to the top of the output file when a calculation commences.

3.9 User-defined basis set ($basis)

The *$rem* variable *BASIS* (Chapter 6) allows the user to indicate that the basis set is being user-defined. The user-defined basis set is entered in the *$basis* section of the input. For further details of entering a user-defined basis set, see chapter 6.

3.10 Geometry Optimization with General Constraints ($opt)

When a user defines the *JOBTYPE* to be a molecular geometry optimization, Q-Chem scans the input deck for the *$opt* keyword. Distance, angle, dihedral and out-of-plane bend constraints imposed on any atom declared by the user in this section, are then imposed on the optimization procedure. See chapter 7 for details.

3.11 USER-DEFINED OCCUPIED GUESS ORBITALS ($OCCUPIED)

It is sometimes useful for the occupied guess orbitals to be other than the lowest Nalpha (or Nbeta) orbitals. Q-Chem allows the occupied guess orbitals to be defined using the *$occupied* keyword. The user defines occupied guess orbitals by listing the alpha orbitals to be occupied on the first line, and beta on the second (see chapter 4).

3.12 NATURAL BOND ORBITAL PACKAGE ($NBO)

The default action in Q-Chem is not to run the NBO package. To turn the NBO package on, set the *$rem* variable *NBO* to *ON*. To access further features of NBO, place standard NBO package parameters into a keyword section in the input file headed with the *$nbo* keyword. Terminate the section with the termination string *$end*.

3.13 ADDITION OF EXTERNAL CHARGES ($EXTERNAL_CHARGES)

If the *$external_charges* keyword is present, Q-Chem scans for a set of external charges to be incorporated into a calculation. The format for a set of external charges is the Cartesian coordinates, followed by the charge size, one charge per line. Charges and coordinates are in atomic units. Coordinates are in the Standard Nuclear Orientation.

```
$external_charges
x-coord1       y-coord1       z-coord1       charge1
x-coord2       y-coord2       z-coord2       charge2
x-coord3       y-coord3       z-coord3       charge3
...
$end
```

Figure 3.8 General format for incorporating s set of external charges.

3.14 APPLYING A MULTIPOLE FIELD ($MULTIPOLE_FIELD)

Q-Chem has the capability to apply a multipole field to the molecule under investigation. Q-Chem scans the input deck for the *$multipole_field* keyword, and reads each line (up to the terminator keyword, *$end*) as a single component of the applied field. The format is:

```
$multipole_field
field_component_1   value_1
field_component_2   value_2
...
$end
```

Figure 3.9 General format for imposing a multipole field.

The *field_component* is simply stipulated using the Cartesian representation *e.g.*, X, Y, Z, XX, XY, YY ... XXX, *etc.*, and the value or size of the imposed field is in atomic units.

3.15 USER-DEFINED EXCHANGE-CORRELATION DENSITY FUNCTIONALS ($XC_FUNCTIONAL)

The *EXCHANGE* and *CORRELATION $rem* variables (Chapter 4) allow the user to indicate that the exchange-correlation density functional will be user-defined. The user defined exchange-correlation is to be entered in the *$xc_functional* part of the input. The format is:

```
$XC_functional
X          exchange_symbol         coefficient
X          exchange_symbol         coefficient
.
.
.
C          correlation_symbol      coefficient
C          correlation_symbol      coefficient
.
.
.
K                                  coefficient
```

Figure 3.10 General form for entering user-defined XC functionals.

Note: Coefficients are real numbers.

3.16 MULTIPLE JOBS IN A SINGLE FILE: Q-CHEM BATCH JOB FILES

It is sometimes useful to place a series of jobs into a single ASCII file. This feature is supported by Q-Chem and is invoked by separating jobs with the string "@@@" on a single line. All output is subsequently appended to the same output file for each job within the file.

Note: The first job will overwrite any existing output file of the same name in the working directory. Restarting the job will also overwrite any existing file.

In general, multiple jobs are placed in a single file for two reasons:
1. To use information from a prior job in a later job
2. To keep projects together in a single file

The "@@@" feature allows these objectives to be met, but the following points should be noted:

- Q-Chem reads all the jobs from the input file on initiation and stores them. The user cannot make changes to the details of jobs which have not been run post command line initiation.
- If any single job fails, Q-Chem proceeds to the next job in the batch file.
- No check is made to ensure that dependencies are satisfied, or that information is consistent (*e.g.*, an optimisation job followed by a frequency job; reading in the new geometry from the optimization for the frequency). No check is made to ensure that the optimization was successful. Similarly, it is assumed that both jobs use the same basis set when reading in MO coefficients from a previous job.
- Scratch files are saved between multi-job/single files runs (*i.e.*, using a batch file with "@@@" separators), but are deleted on completion **unless** a third *qchem* command line argument is supplied (see chapter 2).

Using batch files with the "@@@" separator is clearly most useful for cases relating to point 1 above. The alternative would be to cut and paste output, and/or use a third command line argument to save scratch files between separate runs.

For example, the following input file will optimize the geometry of H_2 at HF/6-31G*, calculate vibrational frequencies at HF/6-31G* using the optimized geometry and the self-consistent MO coefficients from the optimization and, finally, perform a single point energy using the optimized geometry at the MP2/6-311G(d,p) level of theory. Each job will use the same scratch area, reading files from previous runs as instructed.

```
$comment
Optimize H-H at HF/6-31G*
$end

$molecule
0 1
H
H 1 r

r = 1.1
$end

$rem
JOBTYPE        OPT       Optimise the bond length
EXCHANGE       HF
CORRELATION    NONE
BASIS          6-31G*
$end

@@@
$comment
Frequencies of H-H at HF/6-31G*
$end

$molecule
READ
$end

$rem
JOBTYPE        FREQ      Calculate vibrational frequencies
EXCHANGE       HF
CORRELATION    NONE
BASIS          6-31G*
SCF_GUESS      READ      Read the MOs from disk
$end

@@@
$comment
H-H at MP2/6-311G(d,p)//HF/6-31G*
$end

$molecule
READ
$end
```

```
$rem
EXCHANGE        HF
CORRELATION     MP2
BASIS           6-311G(d,p)
$end
```

Example 3.10 Example of using information from previous jobs in a single input file.

Notes: (1) Output is concatenated into the same output file.
(2) Only two arguments are necessarily supplied to the command line interface.

3.17 Q-CHEM TEXT INPUT SUMMARY

- Q-Chem text input file uses a series of **keywords**
- Q-Chem scans the input file for keywords, so they do not have to be placed in any particular order
- Each keyword represents a section of the input file
- Q-Chem reads in data, variables and options from keyword sections
- Each keyword section is terminated with *$end*
- Not all keywords have to be entered, but *$rem* and *$molecule* are compulsory
- Input file is case-insensitive
- Multiple jobs in a single input file are separated by the string “@@@” on a single line

Keyword	Description
$molecule	Signifies the beginning of the molecular coordinate input. Input file requisite
$end	Terminates each keyword section
$rem	Job specification and customization parameters. Input file requisite
$basis	Indicates the beginning of the basis set information for user defined basis sets. (See chapter 6)
$comment	All information placed in the section is incorporated into the Q-Chem output file. All other comments remain as input file user comments and are not read
$external_charges	Section containing external charges and positions
$multipole_field	Section contains details of a multipole field to apply
$nbo	Placing Natural Bond Orbital package options
$occupied	Guess orbitals to be occupied
$opt	Constraint definitions for geometry optimizations
$xc_functional	Details of user-defined DFT exchange-correlation functionals

Table 3.1 Description summary of all Q-Chem input keywords.

3.17.1 KEYWORD FORMAT SUMMARY

Keyword: $molecule

```
Four methods:
1. Z-matrix (Angstroms and degrees)
           $molecule
           {Z-Matrix}
           {blank line, if parameters are being used}
           {Z-matrix parameters, if used}
           $end

2. Cartesian Coordinates (Angstroms)
           $molecule
           {Cartesian coordinates}
           {blank line, if parameter are being used}
           {Coordinate parameters, if used}
           $end
```

```
3. Read from a previous calculation
           $molecule
           READ
           $end

4. Read from a file
           $molecule
           READ     filename
           $end
```

Keyword: $rem

```
$rem
rem_variable   rem_option     [user comment]
...
$end
```

Keyword: $basis

```
$basis
atomic_symbol    0
ang_mom_sym      contraction_K    scaling
exp_1            coeff_1_Lmin     coeff_1_(Lmin+1)   ...    coeff_1_Lmax
exp_2            coeff_2_Lmin     coeff_2_(Lmin+1)   ...    coeff_2_Lmax
exp_3            coeff_3_Lmin     coeff_2_(Lmin+1)   ...    coeff_3_Lmax
.                .                .                  .      .
.                .                .                  .      .
exp_K            coeff_K_Lmin     coeff_K_(Lmin+1)   ...    coeff_K_Lmax
****
atomic_symbol    0
ang_mom_sym      contraction_K    scaling
exp_1            coeff_1_Lmin     coeff_1_(Lmin+1)   ...    coeff_1_Lmax
exp_2            coeff_2_Lmin     coeff_2_(Lmin+1)   ...    coeff_2_Lmax
exp_3            coeff_3_Lmin     coeff_2_(Lmin+1)   ...    coeff_3_Lmax
.                .                .                  .      .
.                .                .                  .      .
exp_K            coeff_K_Lmin     coeff_K_(Lmin+1)   ...    coeff_K_Lmax
****
...
****
$end
```

Keyword: $comment

```
$comment
{User comments - copied to output file}
$end
```

Keyword: $external_charges (atomic units)

```
$external_charges
x-coord1       y-coord1       z-coord1       charge1
x-coord2       y-coord2       z-coord2       charge2
...
$end
```

Keyword: $multipole_field (atomic units)

```
$multipole_field
field_component1    value1
field_component2    value2
...
$end
```

Keyword: $nbo

```
$nbo
{Refer to Chapter 8 and NBO Program manual}
{must set $rem NBO to ON to initiate NBO package}
$end
```

Keyword: $occupied

```
$occupied
αi αj αk αl ... {alpha guess orbitals to be occupied}
βl βm βn βo ... {beta guess orbitals to be occupied}
$end
```

Keyword: $opt (Angstroms and degrees)

```
$opt
CONSTRAINT
stre       atom1    atom2     value
...
bend       atom1    atom2     atom3     value
...
outp       atom1    atom2     atom3     atom4     value
...
tors       atom1    atom2     atom3     atom4     value
...
linc       atom1    atom2     atom3     atom4     value
...
linp       atom1    atom2     atom3     atom4     value
...
ENDCONSTRAINT
FIXED
atom       coordinate_reference
...
ENDFIXED
DUMMY
idum       type      list_length    defining_list
...
ENDDUMMY
CONNECT
atom       list_length   list
...
ENDCONNECT
$end
```

Keyword: $xc_functional

```
$xc_functional
X          exchange_symbol          coefficient
X          exchange_symbol          coefficient
.
.
.
C          correlation_symbol       coefficient
C          correlation_symbol       coefficient
.
.
.
K                                   coefficient
$end
```

3.18 Q-Chem Output File

The Q-Chem output file is the file to which details of the job invoked by the user are printed. The type of information printed to this files depends on the type of job (single point energy, geometry optimisation *etc.*) and the *$rem* variable print levels. The general and default form is as follows:

- User input
- Q-Chem citation
- Molecular geometry in Cartesian coordinates
- Molecular point group, nuclear repulsion energy, number of alpha and beta electrons
- Basis set information (number of functions, shells and function pairs)
- SCF details (method, guess, optimization procedure)
- SCF iterations (for each iteration, energy and DIIS error is reported)
- {depends on job type}
- Molecular orbital symmetries
- Mulliken population analysis
- Cartesian multipole moments
- Job completion

Note: Q-Chem overwrites any existing output files in the working directory when it is invoked with an existing file as the output file parameter.

3.19 Q-Chem Scratch Files

The directory represented by the environment variable *QCSCRATCH* is the location Q-Chem places scratch files it creates on execution. Users may wish to use the information created for subsequent calculations.

CHAPTER 4 GROUND STATE METHODS

4.1 INTRODUCTION

In 1926 Schrödinger [1] combined the wave nature of the electron with the statistical knowledge of the electron *viz.* Heisenberg's Uncertainty Principle [2] to formulate an eigenvalue equation for the total energy of a molecular system. If we focus on stationary states and ignore the effects of relativity, we have the time independent, non-relativistic equation

$$H(\mathbf{R},\mathbf{r})\Psi(\mathbf{R},\mathbf{r}) = E(\mathbf{R})\Psi(\mathbf{R},\mathbf{r}) \tag{4.1}$$

where the coordinates **R** and **r** refer to nuclei and electron position vectors respectively and *H* is the Hamiltonian operator (in atomic units)

$$H = -\frac{1}{2}\sum_{i=1}^{N}\nabla_i^2 - \frac{1}{2}\sum_{A=1}^{M}\frac{1}{M_A}\nabla_A^2 - \sum_{i=1}^{N}\sum_{A=1}^{M}\frac{Z_A}{r_{iA}} + \sum_{i=1}^{N}\sum_{j>i}^{N}\frac{1}{r_{ij}} + \sum_{A=1}^{M}\sum_{B>A}^{M}\frac{Z_A Z_B}{R_{AB}} \tag{4.2}$$

∇^2 is the Laplacian operator

$$\nabla^2 \equiv \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2} \tag{4.3}$$

Z is the nuclear charge, M_A is the ratio of the mass of nucleus *A* to the mass of an electron, $R_{AB} = |\mathbf{R}_A - \mathbf{R}_B|$ is the distance between the A^{th} and B^{th} nucleus, $r_{ij} = |\mathbf{r}_i - \mathbf{r}_j|$ is the distance between the i^{th} and j^{th} electrons, $r_{iA} = |\mathbf{r}_i - \mathbf{R}_A|$ is the distance between the i^{th} electron and A^{th} nucleus, *M* is the number of nuclei and *N* is the number of electrons. *E* is an eigenvalue of *H*, equal to the total energy, and the wave function Ψ, is an eigenfunction of *H*.

Separating the motions of the electrons from that of the nuclei, an idea originally due to Born and Oppenheimer [3], yields the electronic Hamiltonian operator.

$$H_{elec} = -\frac{1}{2}\sum_{i=1}^{N}\nabla_i^2 - \sum_{i=1}^{N}\sum_{A=1}^{M}\frac{Z_A}{r_{iA}} + \sum_{i=1}^{N}\sum_{j>i}^{N}\frac{1}{r_{ij}} \tag{4.4}$$

The solution of the corresponding electronic Schrödinger equation

$$H_{elec}\Psi_{elec} = E_{elec}\Psi_{elec} \tag{4.5}$$

gives the total electronic energy (E_{elec},) and electronic wave function Ψ_{elec}, which describes the motion of the electrons for a fixed nuclear position. The total energy is obtained by simply adding the nuclear-nuclear repulsion energy (fifth term of (4.2)) to the total electronic energy

$$E_{Tot} = E_{elec} + E_{nuc} \tag{4.6}$$

Solving the eigenproblem (4.5) yields a set of eigenfunctions (Ψ_0, Ψ_1, Ψ_2 ...) with corresponding eigenvalues (E_0, E_1, E_2 ...) where $E_0 \leq E_1 \leq E_2 \leq E_3$ ...

Our interest lies in determining the lowest eigenvalue and associated eigenfunction which correspond to the ground state energy and wavefunction of the molecule. However, solving (4.5) for other than the most trivial systems is extremely difficult and the best we can do in practice is to find approximate solutions.

The first approximation used to solve (4.5) is that electrons move independently within molecular orbitals (MO), each of which describes the probability distribution of a single electron. Each MO is determined by considering the electron as moving within an average field of all the other electrons. Ensuring that the wave function is antisymmetric upon electron interchange, yields the well known Slater [4, 5] determinant wavefunction

$$\Psi = \frac{1}{\sqrt{n!}} \begin{vmatrix} \chi_1(1) & \chi_2(1) & \cdots & \chi_n(1) \\ \chi_1(2) & \chi_2(2) & \cdots & \chi_n(2) \\ \vdots & \vdots & & \vdots \\ \chi_1(n) & \chi_2(n) & \cdots & \chi_n(n) \end{vmatrix} \tag{4.7}$$

where χ_i, a spin orbital, is the product of a molecular orbital ψ_i and a spin function (α or β).

One obtains the optimum set of MOs by variationally minimizing the energy in what is called a “self-consistent field” or SCF approximation to the many-electron problem. The archetypal SCF method is the Hartree-Fock approximation, but these SCF methods also include Kohn-Sham Density Functional Theories (see section 4.5). All SCF methods lead to equations of the form

$$f(i)\chi(\mathbf{x}_i) = \varepsilon\chi(\mathbf{x}_i) \tag{4.8}$$

where the Fock operator *f(i)* can be written

$$f(i) = -\frac{1}{2}\nabla_i^2 + \upsilon^{eff}(i) \tag{4.9}$$

Here $\mathbf{x}_i$ are spin and spatial coordinates of the i^{th} electron, χ are the spin orbitals and ν^{eff} is the effective potential “seen” by the i^{th} electron which depends on the spin orbitals of the other electrons. The nature of the effective potential ν^{eff} depends on the SCF methodology and will be elaborated on in further sections.

The second approximation usually introduced when solving (4.5), is the introduction of an Atomic Orbital (AO) basis. AOs (ϕ_μ) are usually combined linearly to approximate the true MOs. There are many standardized, atom-centered basis sets and details of these are discussed in Chapter 6.

After eliminating the spin components in (4.8) and introducing a finite basis,

$$\psi_i = \sum_\mu c_{\mu i} \phi_\mu \tag{4.10}$$

(4.8) reduces to the Roothaan-Hall matrix equation

$$\mathbf{FC} = \boldsymbol{\varepsilon}\mathbf{SC} \tag{4.11}$$

where **F** is the Fock matrix, **C** is a square matrix of molecular orbital coefficients, **S** is the overlap matrix with elements

$$S_{\mu\nu} = \int \phi_\mu(\mathbf{r})\phi_\nu(\mathbf{r})\,\mathrm{d}\,\mathbf{r} \tag{4.12}$$

and $\boldsymbol{\varepsilon}$ is a diagonal matrix of the orbital energies. Generalizing an unrestricted formalism by introducing separate spatial orbitals for α and β spin (4.7) yields the Pople-Nesbet [6] equations

$$\begin{aligned} \mathbf{F}^\alpha \mathbf{C}^\alpha &= \boldsymbol{\varepsilon}^\alpha \mathbf{S}\mathbf{C}^\alpha \\ \mathbf{F}^\beta \mathbf{C}^\beta &= \boldsymbol{\varepsilon}^\beta \mathbf{S}\mathbf{C}^\beta \end{aligned} \tag{4.13}$$

Solving (4.11) or (4.13) yields the restricted or unrestricted finite basis Hartree-Fock approximation. This approximation inherently neglects the instantaneous electron-electron correlations which are averaged out by the SCF procedure, and while the chemistry resulting from HF calculations often offers valuable qualitative insight, quantitative energetics are often poor. In principle, the DFT SCF methodologies are able to capture all the correlation energy (the difference in energy between the HF energy and the true energy). In practice, the best currently available density functionals perform well, but not perfectly and conventional HF-based approaches to calculating the correlation energy are still often required. Of these conventional approaches, second-order Møller-Plesset perturbation theory (MP2) has been shown to represent a useful comprise between accuracy and computational expense for ground state energy calculations. Details of the theory and the Q-Chem implementation are given below.

Types of ground state energy calculations currently available in Q-Chem are summarized in Table 4.1.

Calculation	$rem Variable JOBTYPE
Single point energy (default)	SINGLE_POINT, SP
Force	FORCE
Equilibrium Structure Search	OPTIMIZATION, OPT
Transition Structure Search	TS
Frequency	FREQUENCY, FREQ

Table 4.1 The type of calculation to be run by Q-Chem is controlled by the *$rem* variable JOBTYPE.

4.2 SCF METHODS

In self-consistent field methods, an initial guess is calculated for the MOs and, from this, an average field seen by each electron can be calculated. A new set of MOs can be obtained by solving the Roothaan-Hall (4.11) or Pople-Nesbet (4.13) eigenvalue equations. This procedure is repeated until the new MOs differ negligibly from those of the previous iteration.

Because they often yield acceptably accurate chemical predictions at a reasonable computational cost, self-consistent field methods are the corner stone of most quantum chemical programs and calculations. The formal costs of many SCF algorithms is $O(N^4)$, that is, they grow with the fourth power of the size (N) of the system. This is slower than the growth of the cheapest conventional correlated methods but recent work by Q-Chem, Inc. and its collaborators has dramatically reduced it to $O(N)$, an improvement that now allows SCF methods to be applied to molecules previously considered beyond the scope of *ab initio* [7] treatment.

Q-Chem, Inc. has endeavoured to implement all the latest advances into all the supported SCF methodologies and has been at the forefront of many of the advances (*e.g.*, linear scaling SCF calculations, efficient and reliable implementation of density functionals). Q-Chem now contains the most comprehensive arsenal of efficient SCF algorithms available.

As discussed, SCF methodologies include both the HF and DFT approximations. Each of these methodologies is discussed in the following sections. Extra detail is given to the linear scaling algorithms for large molecule calculations as well as available options for the initial guess and optimising the SCF.

In order to carry out an SCF calculation using Q-Chem, three *$rem* variables need to be set:

• *BASIS*	to specify the basis set (see chapter 6)
• *EXCHANGE*	method for treating Exchange
• *CORRELATION*	method for treating Correlation (defaults to *NONE*)

4.3 Hartree-Fock

4.3.1 Introduction

As with much of the theory underlying modern quantum chemistry, the Hartree-Fock approximation was developed shortly after publication of the Schrödinger equation, but remained a qualitative theory until the advent of the computer. Although the HF approximation tends to yield qualitative chemical accuracy, rather than quantitative information, and is generally inferior to many of the DFT approaches available, it remains as a useful tool in the quantum chemist's toolkit.

Consider once more the Roothaan-Hall equations (4.13) which can be traced back to the integro-differential equation (4.8) where the effective potential ν^{eff} depends on the SCF methodology. In a restricted HF (RHF) formalism, the effective potential can be written as

$$\upsilon^{eff} = \sum_{a}^{N/2} \left[2J_a(1) - K_a(1)\right] - \sum_{A=1}^{M} \frac{Z_A}{r_{1A}} \tag{4.14}$$

where the coulomb and exchange operators are defined as

$$J_a(1) = \int \psi_a^*(2) \frac{1}{r_{12}} \psi_a(2)\,\mathrm{d}\,\mathbf{r_2} \tag{4.15}$$

$$K_a(1)\psi_i(1) = \left[\int \psi_a^*(2) \frac{1}{r_{12}} \psi_i(2)\,\mathrm{d}\,\mathbf{r_2}\right]\psi_a(1) \tag{4.16}$$

respectively. By introducing an atomic orbital basis, we obtain Fock matrix elements

$$F_{\mu\nu} = H_{\mu\nu}^{core} + J_{\mu\nu} - K_{\mu\nu} \tag{4.17}$$

where the core Hamiltonian matrix elements

$$H_{\mu\nu}^{core} = T_{\mu\nu} + V_{\mu\nu} \tag{4.18}$$

consist of kinetic energy elements

$$T_{\mu\nu} = \int \phi_\mu(\mathbf{r}) \left[-\frac{1}{2}\nabla^2\right] \phi_\nu(\mathbf{r})\,\mathrm{d}\,\mathbf{r} \tag{4.19}$$

and nuclear attraction elements

$$V_{\mu\nu} = \int \phi_\mu(\mathbf{r}) \left[-\sum_A \frac{Z_A}{|\mathbf{R_A} - \mathbf{r}|} \right] \phi_\nu(\mathbf{r})\, \mathrm{d}\,\mathbf{r} \tag{4.20}$$

The Coulomb and Exchange elements are given by

$$J_{\mu\nu} = \sum_{\lambda\sigma} P_{\lambda\sigma} (\mu\nu|\lambda\sigma) \tag{4.21}$$

$$K_{\mu\nu} = \frac{1}{2} \sum_{\lambda\sigma} P_{\lambda\sigma} (\mu\lambda|\nu\sigma) \tag{4.22}$$

where the density matrix elements are

$$P_{\mu\nu} = 2 \sum_{a=1}^{N/2} C_{\mu a} C_{\nu a} \tag{4.23}$$

and the two electron integrals are

$$(\mu\nu|\lambda\sigma) = \iint \phi_\mu(\mathbf{r_1}) \phi_\nu(\mathbf{r_1}) \left[\frac{1}{r_{12}} \right] \phi_\lambda(\mathbf{r_2}) \phi_\sigma(\mathbf{r_2})\, \mathrm{d}\,\mathbf{r_1}\, \mathrm{d}\,\mathbf{r_2} \tag{4.24}$$

Note: The formation and utilization of two-electron integrals is a topic central to the overall performance of SCF methodologies. The performance of the SCF methods in new quantum chemistry software programs can be quickly estimated simply by considering the quality of their atomic orbital integrals packages. See the appendix for details of Q-Chem's AOINTS package.

Substituting the matrix element (4.17) back into the Roothaan-Hall equations (4.11) and solving until self-consistency is achieved will yield the Restricted Hartree-Fock energy and wavefunction. Alternatively, one could have adopted the unrestricted form of the wavefunction by defining an alpha and beta density matrix

$$\begin{aligned} P^\alpha_{\mu\nu} &= \sum_{a=1}^{n_\alpha} C^\alpha_{\mu a} C^\alpha_{\nu a} \\ P^\beta_{\mu\nu} &= \sum_{a=1}^{n_\beta} C^\beta_{\mu a} C^\beta_{\nu a} \end{aligned} \tag{4.25}$$

and the total electron density matrix P^T is simply the sum of the alpha and beta density matrices. The unrestricted alpha Fock matrix

$$F^\alpha_{\mu\nu} = H^{core}_{\mu\nu} + J_{\mu\nu} - K^\alpha_{\mu\nu} \tag{4.26}$$

differs from the restricted one only in the exchange contributions where the alpha exchange matrix elements are given by

$$K_{\mu\nu}^{\alpha} = \sum_{\lambda}^{N}\sum_{\sigma}^{N} P_{\lambda\sigma}^{\alpha}\left(\mu\lambda|\nu\sigma\right) \tag{4.27}$$

4.3.2 CASE APPROXIMATION

The Coulomb Attenuated Schrödinger Equation (CASE) [8] approximation follows from the KWIK [9] algorithm in which the Coulomb operator is separated into two pieces

$$\frac{1}{r_{12}} \equiv \frac{\operatorname{erfc}\left(\omega r_{12}\right)}{r_{12}} + \frac{\operatorname{erf}\left(\omega r_{12}\right)}{r_{12}} \tag{4.28}$$

The first of these two terms is singular but short-range and the second is non-singular but long-range. The CASE approximation is applied by smoothly attenuating all occurrences of the Coulomb operator in (4.2) by neglecting the long-range portion of the identity in (4.28). The parameter ω can be used to tune the level of attenuation. Although the total energies from Coulomb attenuated calculations are significantly different from non-attenuated energies, it is found that relative energies, correlation energies and, in particular, wavefunctions, are not.

By virtue of the exponential decay of the attenuated operator, ERIs can be neglected on a proximity basis yielding a rigorous $O(N)$ algorithm for single point energies. CASE may also be applied in geometry optimizations and frequency calculations.

4.3.3 FEATURES

- Unrestricted (UHF), restricted (RHF) and restricted open-shell (ROHF) Hartree-Fock methods
- Analytic energies, first and second derivatives
- AOINTS, Q-Chem's Atomic Orbital INTegral package (see appendix) has been carefully constructed using the PRISM path approach to maximise efficiency and performance. It is not recommended that the user attempt to customize integral formation, other than through memory allocation. Adjusting the level of integral accuracy and the shell-pair cutoff criteria should be done with caution
- CASE approximation
- Linear construction of the Fock matrix (section 4.8)
- Extensive range of basis sets (Chapter 6)
- Extended set of wavefunction analysis tools (Chapter 8)

4.3.4 JOB CONTROL

The following *$rem* variables are the minimum required in order to run Hartree-Fock calculations. The *$rem* variable settings needed in order to run a ground-state single-point energy Hartree-Fock calculation, *$rem EXCHANGE* is set to *HF* and *$rem CORRELATION* is set to *NONE*.

JOBTYPE

Specifies the calculation

VARIABLE:

STRING

DEFAULT:

SP	Single point energy

OPTIONS:

SP	Single point energy
OPT	Geometry Minimization
TS	Transition Structure Search
FREQ	Frequency Calculation
FORCE	Analytical Force calculation

RECOMMENDATION:

Defaults to single point

EXCHANGE

Specifies the exchange level of theory

VARIABLE:

STRING

DEFAULT:

No default

OPTIONS:

HF	Exact (Hartree-Fock)
Slater, S	Slater
Becke, B	Becke
Gill96, Gill	Gill 1996
Becke(EDF1), B(EDF1)	Becke (EDF1)
PW91, PW	Perdew
B3PW91, Becke3PW91, B3P	B3PW91 hybrid
B3LYP, Becke3LYP	B3LYP hybrid
EDF1	EDF1
General, Gen	User defined combination of K, X and C (refer DFT section)

RECOMMENDATION:

HF for Hartree-Fock calculations

CORRELATION

Specifies the correlation level of theory

VARIABLE:

STRING

DEFAULT:

None	No Correlation

OPTIONS:

None	No Correlation
VWN	Vosko-Wilk-Nusair parameterization #5
LYP	Lee-Yang-Parr (LYP)
PW91, PW	GGA91 (Perdew)
LYP(EDF1)	LYP(EDF1) parameterization
Perdew86, P86	Perdew 1986
PZ81, PZ	Perdew-Zunger 1981
Wigner	Wigner
MP2	MP2
CIS(D)	CIS(D) (excited states)

BASIS

Sets the basis sets to be used

VARIABLE:

STRING

DEFAULT:

No default basis set

OPTIONS:

General, Gen	User defined (*$basis* keyword required)
Symbol	Use standard basis sets as per Chapter 6

RECOMMENDATIONS:

Consult literature and reviews to aid your selection

UNRESTRICTED

Controls the use of restricted or unrestricted orbitals

VARIABLE:

LOGICAL

DEFAULT:

FALSE	(Restricted) Closed-shell systems
TRUE	(Unrestricted) Open-shell systems

OPTIONS:

FALSE	Restricted open-shell HF (ROHF)

RECOMMENDATION:

Use default unless ROHF is desired

OMEGA

Controls the degree of attenuation of the Coulomb operator
VARIABLE:
INTEGER
DEFAULT:
No default
OPTIONS:

n	ω = *n*/1000

INTEGRAL_2E_OPR

Determines the two-electron operator
VARIABLE:
INTEGER
DEFAULT:

-2	Coulomb Operator

OPTIONS:

-1	Apply the CASE approximation
-2	Coulomb Operator

4.3.5 Customization

Listed below are a number of useful options to customize a Hartree-Fock calculation. This is only a short summary of the function of these *$rem* variables.

INCORE_INTS_BUFFER

Controls the size of in-core integral storage buffer
VARIABLE:
INTEGER
DEFAULT:
2,000,000 words (1 word = 8 bytes)
OPTIONS:
User defined size; hardware dependent

DIRECT_SCF

Controls direct SCF
VARIABLE:
LOGICAL
DEFAULT:
Determined by program
OPTIONS:

TRUE	Forces direct SCF
FALSE	Do not use direct SCF

RECOMMENDATION:
Use default; direct SCF switches off in-core integrals

THRESH

Cutoff for neglect of two electron integrals. $10^{-\text{THRESH}}$ (*THRESH* ≤ 12)
VARIABLE:
INTEGER
DEFAULT:
8 for single point energies
10 for optimizations and frequency calculations
OPTIONS:
User-defined
RECOMMENDATION:
Should be at least three greater than *SCF_CONVERGENCE*

METECO

Sets the threshold criteria for discarding shell-pairs
VARIABLE:
INTEGER
DEFAULT:
2 Discard shell-pairs below $10^{-\text{THRESH}}$
OPTIONS:
1 Discard shell-pairs fours orders of magnitude below machine precision
2 Discard shell-pairs below $10^{-\text{THRESH}}$
RECOMMENDATION:
Use default

SCF_PRINT

Controls level of output from SCF procedure to Q-Chem output file
VARIABLE:
INTEGER
DEFAULT:
0 Minimal, concise, useful and necessary output
OPTIONS:
0 Minimal, concise, useful and necessary output
1 Level 0 plus component breakdown of SCF electronic energy
2 Level 1 plus density, Fock and MO matrices on each cycle
3 Level 2 plus two-electron Fock matrix components (Coulomb, HF exchange and DFT exchange-correlation matrices) on each cycle
RECOMMENDATIONS:
Proceed with care; can result in *extremely* large output files at level 2 or higher

SCF_FINAL_PRINT

Controls level of output from SCF procedure to Q-Chem output file at the end of the SCF

VARIABLE:

INTEGER

DEFAULT:

0 No extra print out

OPTIONS:

0 No extra print out
1 Orbital Energies only
2 Level 1 plus MOs
3 Level 2 plus Fock and density matrices

4.3.6 Examples

Provided below are examples of Q-Chem input files to run simple ground state, Hartree-Fock single point energy calculations. See the appendix for more examples of Q-Chem input files.

```
$molecule
0 1
O
H1 O OH
H2 O OH H1 HOH

OH = 1.2
HOH = 120.0
$end

$rem
JOBTYPE        SP       Single Point energy
EXCHANGE       HF       Exact HF exchange
CORRELATION    None     No correlation
BASIS          STO-3G   Basis set
$end

$comment
HF/STO-3G water single point calculation
$end
```

Example 4.1 Example Q-Chem input for a single point energy calculation on water. Note that the declaration of the single point *$rem* variable and level of theory to treat correlation are redundant because they are the same as the Q-Chem defaults.

```
$molecule
0,2
3
$end

$rem
EXCHANGE   HF       Hartree-Fock
BASIS      6-311G   Basis set
$end
```

Example 4.2 UHF/6-311G calculation on the Lithium atom. Note that correlation and the job type were not indicated because Q-Chem defaults automatically to no correlation and single point energies. Note also that, since the number of alpha and beta electron differ, MOs default to an unrestricted formalism.

```
$molecule
0,2
3
$end

$rem
EXCHANGE       HF        Hartree-Fock
UNRESTRICTED   FALSE     Restricted MOs
BASIS          6-311G    Basis set
SCF_ALGORITHM  DM        Direct Minimization
$end
```

Example 4.3 ROHF/6-311G calculation on the Lithium atom. Note again that correlation and the job type need not be indicated.

4.4 DENSITY FUNCTIONAL THEORY

4.4.1 INTRODUCTION

In recent years, Density Functional Theory [10] has emerged as an accurate alternative first-principles approach to quantum mechanical molecular investigations. DFT currently accounts for approximately 90% of all quantum chemical calculations being performed, not only because of its proven chemical accuracy, but also because of its relatively cheap computational expense. These two features suggest that DFT is likely to remain a leading method in the quantum chemist’s toolkit well into the next millennium. Q-Chem currently boasts the fastest implemented Density Functional code for large molecule applications allowing researchers to explore molecular systems previously rendered impractical.

DFT is primarily a theory of electronic ground state structures based on the electron density, ρ(**r**), as opposed to the many-electron wavefunction $\Psi(\mathbf{r_1},...,\mathbf{r_N})$. There are a number of distinct similarities and differences to traditional wavefunction approaches and modern DFT methodologies. Firstly, the essential building blocks of the many electron wavefunction are single-electron orbitals are directly analogous to the Kohn-Sham (see below) orbitals in the current DFT framework. Secondly, both the electron density and the many-electron wavefunction tend to be constructed *via* a SCF approach that requires the construction of matrix elements which are remarkably and conveniently very similar.

However, traditional approaches using the many electron wavefunction as a foundation must resort to a post-SCF calculation to incorporate correlation effects, whereas DFT approaches do not. Post-SCF methods, such as perturbation theory or configuration interaction are extremely expensive relative to the SCF procedure, and do not capture all the correlation, except in the case of full expansions. On the other hand, the DFT approach is, in principle, exact, but relies on the knowledge of an exact exchange correlation energy functional. While more accurate forms of such functionals are constantly being developed, there is no systematic way to improve the functional to achieve an arbitrary level of accuracy. Thus, the traditional approaches offer the possibility of achieving an arbitrary level of accuracy, but can be computationally demanding, whereas DFT approaches offer a practical route but the theory is currently incomplete.

4.5 KOHN-SHAM DENSITY FUNCTIONAL THEORY

The Density Functional Theory by Hohenberg, Kohn and Sham [10-12] stems from the original work of Dirac [13], who found that the exchange energy of a uniform electron gas may be calculated exactly, knowing only the charge density. However, while the more traditional DFT constitutes a direct approach and the necessary equations contain only the electron density, difficulties associated with the kinetic energy functional obstructed the extension of DFT to anything more than a crude level of approximation. Kohn and Sham developed an indirect approach to the kinetic energy functional which transformed DFT into a practical tool for quantum chemical calculations.

Within the Kohn-Sham formalism [10-11], the ground state electronic energy, *E*, can be written as

$$E = E_T + E_V + E_J + E_{XC} \tag{4.29}$$

where E_T is the kinetic energy, E_V is the electron-nuclear interaction energy, E_J is the Coulomb self-interaction of the electron density ρ(**r**) and E_{XC} is the exchange-correlation energy. Adopting an unrestricted format, the alpha and beta total electron densities can be written as

$$\begin{aligned}\rho_\alpha(\mathbf{r}) &= \sum_{i=1}^{n_\alpha}\left|\psi_i^\alpha\right|^2 \\ \rho_\beta(\mathbf{r}) &= \sum_{i=1}^{n_\beta}\left|\psi_i^\beta\right|^2\end{aligned} \tag{4.30}$$

where n_α and n_β are the number of alpha and beta electron respectively and, ψ_i are the Kohn-Sham orbitals. Thus, the total electron density is

$$\rho(\mathbf{r}) = \rho_\alpha(\mathbf{r}) + \rho_\beta(\mathbf{r}) \tag{4.31}$$

which within a finite basis [14] is represented by

$$\rho(\mathbf{r}) = \sum_{\mu\nu} P_{\mu\nu}^T \phi_\mu(\mathbf{r})\phi_\nu(\mathbf{r}) \tag{4.32}$$

The components of (4.29) can now be written as

$$\begin{aligned}E_T &= \sum_{i=1}^{n_\alpha}\left\langle\psi_i^\alpha\right|-\frac{1}{2}\nabla^2\left|\psi_i^\alpha\right\rangle + \sum_{i=1}^{n_\beta}\left\langle\psi_i^\beta\right|-\frac{1}{2}\nabla^2\left|\psi_i^\beta\right\rangle \\ &= \sum_{\mu\nu} P_{\mu\nu}^T\left\langle\phi_\mu(\mathbf{r})\right|-\frac{1}{2}\nabla^2\left|\phi_\nu(\mathbf{r})\right\rangle\end{aligned} \tag{4.33}$$

$$\begin{aligned}E_V &= -\sum_{A=1}^{M} Z_A \frac{\rho(\mathbf{r})}{\left|\mathbf{r}-\mathbf{R_A}\right|}\mathrm{d}\,\mathbf{r} \\ &= -\sum_{\mu\nu} P_{\mu\nu}^T \sum_A \left\langle\phi_\mu(\mathbf{r})\right|\frac{Z_A}{\left|\mathbf{r}-\mathbf{R_A}\right|}\left|\phi_\nu(\mathbf{r})\right\rangle\end{aligned} \tag{4.34}$$

$$\begin{aligned}E_J &= \frac{1}{2}\left\langle\rho(\mathbf{r_1})\right|\frac{1}{\left|\mathbf{r_1}-\mathbf{r_2}\right|}\left|\rho(\mathbf{r_2})\right\rangle \\ &= \frac{1}{2}\sum_{\mu\nu}\sum_{\lambda\sigma} P_{\mu\nu}^T P_{\mu\nu}^T (\mu\nu|\lambda\sigma)\end{aligned} \tag{4.35}$$

$$E_{XC} = \int f\left(\rho(\mathbf{r}), \nabla\rho(\mathbf{r}), \ldots\right)\mathrm{d}\,\mathbf{r} \tag{4.36}$$

Minimizing E with respect to the unknown Kohn-Sham orbital coefficients yields a set of matrix equations exactly analogous to the UHF case

$$\begin{aligned}\mathbf{F}^\alpha\mathbf{C}^\alpha &= \boldsymbol{\varepsilon}^\alpha\mathbf{S}\mathbf{C}^\alpha \\ \mathbf{F}^\beta\mathbf{C}^\beta &= \boldsymbol{\varepsilon}^\beta\mathbf{S}\mathbf{C}^\beta\end{aligned} \tag{4.37}$$

where the Fock matrix elements are generalised to

$$
\begin{aligned}
F_{\mu\nu}^{\alpha} &= H_{\mu\nu}^{core} + J_{\mu\nu} - F_{\mu\nu}^{XC\alpha} \\
F_{\mu\nu}^{\beta} &= H_{\mu\nu}^{core} + J_{\mu\nu} - F_{\mu\nu}^{XC\beta}
\end{aligned}
\tag{4.38}
$$

where $F_{\mu\nu}^{XC\alpha}$ and $F_{\mu\nu}^{XC\beta}$ are the exchange-correlation parts of the Fock matrices dependent on the exchange-correlation functional used. The Pople-Nesbet equations are obtained simply by allowing

$$
F_{\mu\nu}^{XC\alpha} = K_{\mu\nu}^{\alpha} \tag{4.39}
$$

and similarly for the beta equation.

Thus, the density and energy are obtained in a manner analogous to that for the Hartree-Fock method. Initial guesses are made for the MO coefficients and an iterative process applied until self consistency is obtained.

4.6 EXCHANGE-CORRELATION FUNCTIONALS

There are an increasing number of exchange and correlation functionals and hybrid DFT methods available to the quantum chemist, many of which are very effective. In short, there are two basic types of functionals: those based on the local spin density approximation (LSDA) and those based on the generalized gradient approximation (GGA). Explicit definitions of each of these approximations vary amongst theoreticians and the reader is referred to the literature for further details.

Q-Chem includes the following LSDA functionals:

- Slater-Dirac (Exchange) [13]
- Vokso-Wilk-Nusair (Correlation) [15]
- Perdew-Zunger (Correlation) [16]
- Wigner (Correlation) [17]

the following GGA functionals

- Becke88 (Exchange) [18]
- Gill96 (Exchange) [19]
- Lee-Yang-Parr (Correlation) [20]
- Perdew86 (Correlation) [21]
- GGA91 (Exchange and correlation) [22]

and the pure DFT EDF1 [31] exchange-correlation functional recently reported by Adamson e*t al*. EDF1 contains no HF exchange and when used with the 6-31+G* basis set, is more accurate than B3LYP [32] and B3PW91 [30] hybrids (next section).

4.6.1 Hybrid Functionals

Hybrid exchange-correlation functionals [30], whereby several different exchange and correlation functionals are combined linearly to form a new functional, have proven successful in a number of reported applications. However, since Hybrid functionals contain HF exchange they are more expensive that pure DFT functionals. Q-Chem has incorporated two the hybrid functionals B3LYP [32] and B3PW91 [30] with the additional option for users to define their own hybrid functionals *via* the *$xc_functional* keyword (see customization, below).

Note: The hybrid functionals are not simply a pairing of an exchange and correlation functional, but are a combined exchange-correlation functional (*i.e.*, B-LYP and B3LYP vary in the correlation contribution in addition to the exchange part).

4.7 DFT Numerical Quadrature

In practical DFT calculations, the forms of the approximate exchange-correlation functionals (4.36) used are quite complicated, such that the required integrals involving the functionals generally cannot be evaluated analytically. Q-Chem evaluates these integrals through numerical quadrature directly applied to the exchange-correlation integrand (*i.e.*, no fitting of the XC potential in an auxiliary basis is done).

The quadrature approach in Q-Chem is generally similar to that found in many DFT programs. The multicenter XC integrals are first partitioned into "atomic" contributions using a nuclear weight function. Q-Chem uses the nuclear partitioning of Becke [23], though without the "atomic size adjustments". The atomic integrals are then evaluated through standard one-center numerical techniques.

Thus, the exchange-correlation energy E_{XC} (4.36) is obtained as

$$E_{XC} = \sum_A \sum_i w_{Ai} f\left(\mathbf{r}_{Ai}\right) \qquad (4.40)$$

where the first summation is over the atoms and the second is over the numerical quadrature grid points for the current atom. The *f* function is the exchange-correlation functional. The w_{Ai} are the quadrature weights, and the grid points $\mathbf{r}_{Ai}$ are given by

$$\mathbf{r}_{Ai} = \mathbf{R}_A + \mathbf{r}_i \qquad (4.41)$$

where $\mathbf{R}_A$ is the position of nucleus *A*, with the $\mathbf{r}_i$ defining a suitable one-centre integration grid, which is independent of the nuclear configuration.

The single-centre integrations are further separated into radial and angular integrations. The radial part is treated by the Euler-Maclaurin scheme (see later) and the angular part treated by Lebedev and Gauss-Legendre formulae.

Q-Chem provides a standard quadrature grid by default which is sufficient for most purposes.

4.7.1 Radial Grids

Q-Chem uses Euler-Maclaurin radial grids. This method, proposed by Handy[24], is based on the Euler-Maclaurin formula for summation of a series.

4.7.2 Angular Grids

Angular quadrature rules may be characterized by their degree, which is the highest degree of spherical harmonics for which the formula is exact, and their efficiency, which is the number of spherical harmonics exactly integrated per degree of freedom in the formula. Q-Chem supports the following types of angular grids:

- **Lebedev**

These are specially constructed grids for quadrature on the surface of a sphere [25-27] based on the octahedral group. Lebedev grids of the following degrees are available:

  - 3rd degree, 6 points
  - 5th degree, 18 points
  - 7th degree, 26 points
  - 9th degree, 38 points
  - 11th degree, 50 points
  - 15th degree, 86 points
  - 17th degree, 110 points
  - 19th degree, 146 points
  - 23rd degree, 194 points
  - 29th degree, 302 points

Lebedev grids typically have efficiencies near one, with efficiencies greater than one in some cases.

- **Gauss-Legendre**

These are spherical product rules separating the two angular dimensions θ and ϕ. Integration in the θ dimension is carried out with a Gaussian quadrature rule derived from the Legendre polynomials (orthogonal on [-1,1] with weight function unity), while the ϕ integration is done with equally spaced points.

A Gauss-Legendre grid is selected by specifying the total number of points, $2N_\theta^2$, to be used for the integration. This gives a grid with $2N_\theta$ ϕ-points, N_θ θ–points, and a degree of $2N_\theta$-1.

In contrast with Lebedev grids, Gauss-Legendre grids have efficiency of only 2/3 (hence more Gauss-Legendre points are required to attain the same accuracy as Lebedev). However, since Gauss-Legendre grids of general degree are available, this is a convenient mechanism for achieving arbitrary accuracy in the angular integration if desired.

4.7.3 STANDARD QUADRATURE GRID

The default grid used in Q-Chem 1.2 is the SG-1 standard quadrature grid [28]. This grid was designed to yield the performance of a large, accurate quadrature grid, but with as few points as possible for the sake of computational efficiency. This is accomplished by reducing the number of angular points in regions where sophisticated angular quadrature is not necessary, such as near the nuclei where the charge density is nearly spherically symmetric, while retaining large numbers of angular points in the valence region where angular accuracy is critical.

The SG-1 grid is derived in this fashion from a Euler-Maclaurin-Lebedev-(50,194) grid (*i.e.*, 50 radial points, and 194 angular points per radial point). This grid has been found to give numerical integration errors of the order of 0.2 kcal/mol for medium-sized molecules, including particularly demanding test cases such as isomerization energies of alkanes. This error is deemed acceptable since it is significantly smaller than the accuracy typically achieved by quantum chemical methods. In SG-1 the total number of points is reduced to approximately 1/4 of that of the original EML-(50,194) grid, with SG-1 generally giving the same total energies as EML-(50,194) to within a few microhartrees (0.01 kcal/mol). Therefore, the SG-1 grid is relatively efficient while still maintaining the numerical accuracy necessary for chemical reliability in the majority of applications.

4.7.4 INTERNAL CONSISTENCY CHECK ON NUMERICAL INTEGRATION

Whenever Q-Chem calculates numerical density functional integrals, the electron density itself is also integrated numerically as a test on the quality of the quadrature formula used. The deviation of the numerical result from the number of electrons in the system is an indication of the accuracy of the other numerical integrals. If the relative error in the numerical electron count reaches 0.01%, a warning is printed; this is an indication that the numerical XC results may not be reliable. If the warning appears at the first SCF cycle, it is probably not serious, because the initial-guess density matrix is sometimes not idempotent, as is the case with the SAD guess and the density matrix taken from a different geometry in a geometry optimization. If that is the case, the problem will be corrected as the idempotency is restored in later cycles. On the other hand, if the warning is persistent to the end of SCF iterations, then either a finer grid is needed, or choose an alternative method for generating the initial guess.

Users should be aware, however, of the potential flaws that have been discoverd in some of the grids currently in use. Jarecki and Davidson [57], for example, have recently shown that correctly integrating the density is a necessary, but not sufficient, test of grid quality.

4.7.5 Exchange-Correlation Cutoffs

By default, Q-Chem will estimate the magnitude of various XC contributions on the grid and eliminate those determined to be numerically insignificant. Q-Chem uses specially developed cutoff procedures which permits evaluation of the XC energy and potential in only $O(N)$ work for large molecules, where N is the size of the system. This is a significant improvement over the formal $O(N^3)$ scaling of the XC cost, and is critical in enabling DFT calculations to be carried out on very large systems. In very rare cases, however, the default cutoff scheme can be too aggressive, eliminating contributions that should be retained; this is almost always signalled by an inaccurate numerical density integral. An example of when this could occur is in calculating anions with multiple sets of diffuse functions in the basis. As mentioned above, when an inaccurate electron count is obtained, it maybe possible to remedy the problem by increasing the size of the quadrature grid.

4.7.6 Rotational invariance and Standard Nuclear Orientation

Early implementations of quadrature-based Kohn-Sham DFT employing standard basis sets were plagued by lack of rotational invariance. That is, rotation of the system yielded a significantly energy. Clearly, such behaviour is highly undesirable.

Johnson *et al.* rectified the problem of rotational invariance by completing the specification of the grid procedure [29] to ensure that the computed XC energy is the same for any orientation of the molecule in any Cartesian coordinate system.

4.7.7 Features

Q-Chem contains numerous important and widely adopted DFT features which can be grouped generally as follows:

- Local Spin Density Approximation (LSDA) functionals
- Generalised Gradient Approximations (GGA) Functionals
- Standard and user-defined hybrid functionals
- Standard and user-defined grids
- Internal quadrature consistency checks and cutoffs

4.7.8 JOB CONTROL

The following *$rem* variables are required to run DFT a calculation.

EXCHANGE

Specifies the exchange functional or exchange-correlation functional for hybrids

VARIABLE:

STRING

DEFAULT:

No default exchange functional

OPTIONS:

HF	Exact (Hartree-Fock)
Slater, S	Slater
Becke, B	Becke
Gill96, Gill	Gill 1996
Becke(EDF1), B(EDF1)	Becke (EDF1)
PW91, PW	Perdew
B3PW91, Becke3PW91, B3P	B3PW91 hybrid
B3LYP, Becke3LYP	B3LYP hybrid
EDF1	EDF1
General, Gen	User defined combination of K, X and C (refer next section)

RECOMMENDATION:

DFT exchange functional or hybrid exchange-correlation functional as required by the user.

CORRELATION

Specifies the correlation functional

VARIABLE:

STRING

DEFAULT:

None No correlation

OPTIONS:

None	No correlation
VWN	Vosko-Wilk-Nusair parameterization #5
LYP	Lee-Yang-Parr (LYP)
PW91, PW	GGA91 (Perdew)
LYP(EDF1)	LYP(EDF1) parameterization
Perdew86, P86	Perdew 1986
PZ81, PZ	Perdew-Zunger 1981
Wigner	Wigner
MP2	MP2
CIS(D)	CIS(D) (excited states)

RECOMMENDATION:

DFT correlation functional as required by the user. Note that for hybrid and user-defined schemes the correlation setting is not required

BASIS

Sets the basis sets to be used

VARIABLE:

STRING

DEFAULT:

No default basis set

OPTIONS:

General, Gen	User defined. (*$basis* keyword required)
Symbol	Use standard basis sets as per Chapter 6

RECOMMENDATIONS:

Consult literature and reviews to aid your selection

XC_GRID

Specifies the type of grid to use for DFT calculations.

DEFAULT:

1	SG-1

OPTIONS:

1	SG-1
2	Low Quality
mn	The first six integers correspond to *m* radial points and the second six integers correspond to *n* angular points where possible numbers of Lebedev angular points are listed in section 4.7.2
-*mn*	The first six integers correspond to *m* radial points and the second six integers correspond to *n* angular points where the number of Gauss-Legendre angular points $n = 2N_\theta^2$

RECOMMENDATION:

SG-1

4.7.9 User-Defined Density Functionals

The format for entering user-defined exchange-correlation density functionals is one line for each component of the functional. Each line requires three variables: the first defines whether the component is an exchange or correlation functional by declaring an *X* or *C*, respectively. The second variable is the symbolic representation of the functional as used for the *EXCHANGE* and *CORRELATION $rem* variables. The final variable is a real number corresponding to the contribution of the component to the functional. Hartree-Fock exchange contributions (required for hybrid density functionals) can be entered using only two variables (*K*, for HF exchange) followed by a real number.

```
$XC_functional
X          exchange_symbol         coefficient
X          exchange_symbol         coefficient
.
.
.
C          correlation_symbol      coefficient
C          correlation_symbol      coefficient
.
.
.
K                                  coefficient
$end
```

Notes: (1) Coefficients are real.
(2) A user-defined functional does not require all *X*, *C* and *K* components.

4.7.10 Examples

```
$comment
B-LYP/STO-3G water single point calculation
$end

$molecule
0 1
O
H1 O OH
H2 O OH H1 HOH

OH = 1.2
HOH = 120.0
$end

$rem
EXCHANGE       Becke          Becke88 exchange
CORRELATION    LYP            LYP correlation
BASIS          STO-3G         Basis set
$end
```

Example 4.4 Example Q-Chem input for a DFT single point energy calculation on water

```
$comment
EDF1/6-31+G* water single point calculation
$end

$molecule
0 1
O
H1 O OH
H2 O OH H1 HOH

OH = 1.2
HOH = 120.0
$end

$rem
EXCHANGE       EDF1      EDF1 exchange-correlation
BASIS          6-31+G*   Basis set
$end
```

Example 4.5 Example Q-Chem input for a DFT single point energy calculation on water

4.8 LARGE MOLECULE OPTIONS

4.8.1 INTRODUCTION

Q-Chem’s superior program design coupled with numerous collaborations with world-leading researchers in the area of large molecule calculations enabled Q-Chem to routinely treat large molecular systems consisting of hundreds of atoms (several thousand basis functions). These are systems that were beyond the realm of possibility just a few years ago.

The quantum chemical Coulomb problem, perhaps better known as the DFT bottleneck, has been at the forefront of many research efforts throughout the 1990s. The quadratic computational scaling behaviour seen in the construction of the Coulomb matrix in DFT calculations, and, additionally, the exchange matrix in Hartree-Fock calculations, has prevented the application of *ab initio* methods to molecules containing many hundreds of atoms. Q-Chem, Inc., in collaboration with White and Head-Gordon at the University of California at Berkeley, and Gill at Massey University in New Zealand, were the first to develop the generalization of Greengard’s Fast Multipole Method (FMM) to Continuous charged matter distributions in the form of the CFMM, which is the first linear scaling algorithm for DFT calculations. This initial breakthrough has since lead to an increasing number of linear scaling alternatives and analogues [8, 9].

Within Q-Chem are a number of linear scaling algorithms including the CFMM, QCTC, ONX, CASE and DFT exchange and correlation implementations. However, these algorithms are not the only approaches the user can take to speeding up calculations on large molecules.

4.8.2 Incremental and Variable Thresh Fock Matrix Building

The use of a variable integral threshold, operating for the first few cycles of an SCF, is justifiable on the basis that the MO coefficients are usually of poor quality in these cycles. In Q-Chem, the integrals in the first iteration are calculated at a threshold of 10^{-6} (for an anticipated final integral threshold greater than, or equal to 10^{-6}) to ensure the error in the first iteration is solely sourced from the poor MO guess. Following this, the integral thresh-hold used is computed as

$$tmp\_thresh = varthresh \times DIIS\_error \tag{4.42}$$

where the *DIIS_error* is that calculated from the previous cycle, *varthresh* is the variable threshold set by the program (by default) and *tmp_thresh* is the temporary threshold used for integral evaluation. Each cycle requires recalculation of all integrals. The variable integral threshold procedure has the greatest impact in early SCF cycles.

In an incremental Fock matrix build [33], **F** is computed recursively as

$$\mathbf{F}^{\mathbf{m}} = \mathbf{F}^{\mathbf{m-1}} + \Delta\mathbf{J}^{\mathbf{m-1}} - \tfrac{1}{2}\Delta\mathbf{K}^{\mathbf{m-1}} \tag{4.43}$$

where *m* is the SCF cycle, and $\Delta\mathbf{J}^{\mathrm{m}}$ and $\Delta\mathbf{K}^{\mathrm{m}}$ are computed using the difference density

$$\Delta\mathbf{P}^{\mathbf{m}} = \mathbf{P}^{\mathbf{m}} - \mathbf{P}^{\mathbf{m-1}} \tag{4.44}$$

Using Schwartz integrals and elements of the difference density, Q-Chem is able to determine at each iteration which ERIs are required, and if necessary, recalculated. As the SCF nears convergence, $\Delta\mathbf{P}^{\mathrm{m}}$ becomes sparse and the number of ERIs that need to be recalculated declines dramatically, saving the user large amounts of computational time.

Incremental Fock matrix builds and variable thresholds are only used when the SCF is carried out using the direct SCF algorithm and are clearly complementary algorithms.

4.8.3 CFMM

The Continuous Fast Multipole Method was the first implemented linear scaling algorithm for the construction of the **J** matrix. In collaboration with Q-Chem, Inc., Dr. Chris White began the development of the CFMM by more efficiently deriving [34] the original Fast Multipole Method [35] before generalizing to CFMM [36]. The generalization applied by White *et al.* allowed the principles underlying the success of the FMM to be applied to *arbitrary* (subject to constraints in evaluating the related integrals) continuous, but localized, matter distributions. White e*t al.* further improved the underlying CFMM algorithm [38, 39] then implemented it efficiently [37], achieving performance that is an order of magnitude faster than some competing implementations.

The success of the CFMM follows similarly with that of the FMM, in that the charge system is subdivided into a hierarchy of boxes. Local charge distributions are then systematically organized into multipole representations so that each distribution interacts with local expansions of the potential due to all distant charge distributions. Local and distant distributions are distinguished by a well-separated (WS) index, which is the number of boxes which must separate two collections of charges before they may be considered distant and can interact through multipole expansions; near-field interactions must be calculated directly. In the CFMM each distribution is given its own WS index and is sorted on the basis of the WS index, and the position of their space centres. The implementation in Q-Chem has allowed the efficiency gains of contracted basis functions to be maintained.

The CFMM algorithm can be summarised in five steps:

1. Form and translate multipoles.
2. Convert multipoles to local Taylor expansions.
3. Translate Taylor information to the lowest level.
4. Evaluate Taylor expansions to obtain the far-field potential.
5. Perform direct interactions between overlapping distributions.

Accuracy can be carefully controlled by due consideration of tree depth, truncation of the multipole expansion and the definition of the extent of charge distributions in accordance with a rigorous mathematical error bound. As a rough guide, 10 poles are adequate for single point energy calculations, while 25 poles yield sufficient accuracy for gradient calculations. Subdivision of boxes to yield a one-dimensional length of about 8 boxes works quite well for systems of up to about one hundred atoms. Larger molecular systems, or ones which are extended along one dimension, will benefit from an increase in this number. The program automatically selects an appropriate number of boxes by default.

4.8.4 LINEAR FOCK MATRIX CONSTRUCTION *VIA* QCTC/ONX

The Quantum Chemical Tree Code [40, 41] has been a highly successful addition to the now increasing number of hierarchical multipole expansion methods available [34, 36, 37, 42-45]. Unlike the CFMM, which is a fast multipole method, QCTC is a tree code. While the two groupings of hierarchical multipole expansion methods have the common feature of recursive subdivision of the charge system into a hierarchy of cells, they differ in that the tree codes involve particle-cell interactions whilst the fast multipole methods use cell-cell multipole-to-local transformations of the potential. Other differences include the width of the error estimates, which are tighter for tree codes, and the perceived computational cost scaling characteristics.

The current implementation of QCTC can compute relevant Fock matrix elements up to angular momentum *d*. The QCTC algorithm, which computes **J**, quite rapidly becomes competitive with efficient quadratic ERI code with increasing molecule size. The ONX algorithm, which computes *K*, is less competitive for small to medium sized systems, but is quite beneficial for large systems. (The Q-Chem web site URL: http://www.q-chem.com contains benchmark calculations to quantify these comments.)

4.8.5 FEATURES

- Linear Scaling construction of the Coulomb matrix *via* the CFMM or QCTC
- Linear Scaling construction of the Hartree-Fock Fock matrix *via* CFMM, QCTC and ONX
- Fastest implementation of the CFMM currently available
- The only rigorously linear scaling algorithm for the construction of the Hartree-Fock Exchange matrix
- Incremental Fock matrix build
- Variable integral threshold for early SCF cycles
- CASE approximation (see HF section)

4.8.6 JOB CONTROL

To exploit Q-Chem's large molecule features the user must supply the details of the theoretical procedure in the *$rem* keyword.

4.8.7 CFMM $REM VARIABLES

CFMM_ORDER

Controls the order of the multipole expansions in CFMM calculation

VARIABLE:

INTEGER

DEFAULT:

15	For single point SCF accuracy
25	For tighter convergence

OPTIONS:

n	Use multipole expansions of order *n*

RECOMMENDATION:

Use default

GRAIN

Controls the number of lowest-level boxes in one dimension for CFMM
VARIABLE:
INTEGER
DEFAULT:
-1 Program decides best value, turning on CFMM when useful
OPTIONS:
-1 Program decides best value, turning on CFMM when useful
1 Do not use CFMM
n≥8 Use CFMM with *n* lowest-level boxes in one dimension
RECOMMENDATIONS:
This is an expert option; either use the default, or use a value of 1 if CFMM is not desired

4.8.8 VARIABLE THRESH AND INCREMENTAL FOCK MATRIX $REM VARIABLES

Note: These options are only used with direct SCF calculations.

INCFOCK

Iteration number after which the incremental Fock matrix algorithm is initiated
VARIABLE:
INTEGER
DEFAULT:
1 Start INCFOCK after iteration number 1
OPTIONS:
User-defined (0 switches INCFOCK off)
RECOMMENDATIONS:
May be necessary to allow several iterations before switching on INCFOCK

VARTHRESH

Controls the temporary integral cut-off threshold.
tmp_thresh = $10^{-\mathrm{VARTHRESH}}$ X DIIS_error
VARIABLE:
INTEGER
DEFAULT:
3
OPTIONS:
User-defined threshold
RECOMMENDATIONS:
3 has been found to be a practical level

4.8.9 ONX/QCTC $REM VARIABLES

ONX

Switch for ONX linear scaling algorithm for Hartree-Fock Exchange
VARIABLE:
    LOGICAL
DEFAULT:
    False   Do not use ONX
OPTIONS:
    True   Use ONX (requires CFMM or QCTC)
    False   Do not use ONX

QCTC

Switch for the Quantum Chemical Tree Code linear scaling algorithm for Coulomb interactions
VARIABLE:
    LOGICAL
DEFAULT:
    False   Do not use QCTC
OPTIONS:
    True   Use QCTC
    False   Do not use QCTC

4.8.10 CUSTOMIZATION

Q-Chem's integral storage manager considers the defined level of memory available for integral storage (*INCORE_INTS_BUFFER*; default 2000000) and the overall memory requirement for the total number of integrals. Large calculations will very likely result in the size of the buffer being insignificant compared to the total number of integrals, and, in such cases, Q-Chem will revert to the direct SCF integrals evaluation algorithm.

DIRECT_SCF

Controls direct SCF
VARIABLE:
    LOGICAL
DEFAULT:
    Determined by program
OPTIONS:
    TRUE   Forces direct SCF
    FALSE   Do not use direct SCF
RECOMMENDATION:
    Use default; *DIRECT_SCF* switches off in-core integrals

The primary customization feature of ONX and QCTC is memory allocation. The following *$rem* variables do not correspond to total memory usage, but to variables used in memory allocation. Thus, these variables should be increased with care.

ONX_MXDIS

Memory allocation control variable for ONX
VARIABLE:
INTEGER
DEFAULT:
50,000 (words)
OPTIONS:
User defined.
RECOMMENDATIONS:
May require increasing for large jobs

ONX_MXPRM

Memory allocation control variable for ONX.
VARIABLE:
INTEGER
DEFAULT:
100,000 (words)
OPTIONS:
User defined.
RECOMMENDATIONS:
May require increasing for large jobs

QCTC_MAXBIGMEM

Memory allocation control variable for QCTC
VARIABLE:
INTEGER
DEFAULT:
300,000 (words)
OPTIONS:
User defined
RECOMMENDATIONS:
May require increasing for large jobs.
Note: may be necessary especially for crashes in *maketree* and *initrho* subroutines.

Error tolerances are controlled by the same variable as that in AOINTS - *THRESH*. The QCTC uses a tolerance for cutoffs on the basis of two orders of magnitude below *THRESH*.

4.8.11 Examples

```
$comment
HF/3-21G single point calculation on a large molecule
read in the molecular coordinates from file
$end

$molecule
READ       dna.inp
$end

$rem
EXCHANGE       HF        HF exchange
BASIS          3-21G     Basis set
QCTC           TRUE      Calculate J using QCTC
ONX            TRUE      Calculate K using ONX
$end
```

Example 4.6 Example Q-Chem input for a large single point energy calculation

```
$comment
HF/3-21G single point calculation on a large molecule
read in the molecular coordinates from file
$end

$molecule
READ       dna.inp
$end

$rem
EXCHANGE       HF        HF exchange
BASIS          3-21G     Basis set
INCFOCK        5         Incremental Fock after 5 cycles
VARTHRESH      3         1.0d-03 variable threshold
$end
```

Example 4.7 Example Q-Chem input for a large single point energy calculation.

4.8.12 Symmetry

Symmetry is a powerful branch of mathematics and is often exploited in quantum chemistry, both to reduce the computational workload and to classify the final results obtained [46-48]. Q-Chem is able to determine the point group symmetry of the molecular nuclei and, on competition of the SCF procedure, classify the symmetry of molecular orbitals, and provide symmetry decomposition of kinetic and nuclear attraction energy (see Chapter 8).

Molecular systems possessing point group symmetry offer the possibility of large savings of computational time, by avoiding calculations of integrals which are equivalent. *i.e.*, those integrals which can be mapped on to one another under one of the symmetry operations of the molecular point group.

The Q-Chem default is to use symmetry to reduce computational time, when possible. Some algorithms, such as the CFMM, do not yet have symmetry efficiencies implemented and these cases the symmetry flag (*$rem* variable *SYMMETRY*) is ignored.

SYMMETRY

Controls the use of efficiency through the use of point group symmetry
VARIABLE:
LOGICAL
DEFAULT:

TRUE	Use symmetry when available

OPTIONS:

TRUE	Use symmetry when available
FALSE	Do not use symmetry

RECOMMENDATION:
Use default unless benchmarking

4.9 SCF Initial Guess

4.9.1 Introduction

The Roothaan-Hall (4.11) and Pople-Nesbet (4.13) equations are non-linear in the molecular orbital coefficients. Like many mathematical problems involving non-linear equations, prior to the application of a technique to search for a numerical solution, an initial guess for the solution must be generated. If the guess is poor, the iterative procedure applied to determine the numerical solutions may converge very slowly, requiring a large number of iterations, or at worst, the procedure may diverge.

In an *ab initio* SCF procedure, the quality of the initial guess is of utmost importance to ensure that the SCF procedure converges to an appropriate ground state. When considering jobs with many basis functions requiring the recalculation of ERIs at each iteration, using an intelligent guess mechanism can reduce the total job time significantly.

Q-Chem currently offer four options for the initial guess:

- Superposition of Atomic Density (SAD)
- Core Hamiltonian
- Generalized Wolfsberg-Helmholtz (GWH)
- Reading previously obtained MOs from disk.

4.9.2 SAD

The SAD guesser uses a Superposition of Atomic Densities to construct a density matrix guess. The SAD guesser has been found to be superior to techniques traditionally found in quantum chemistry software and hence, is the default.

Note: The SAD guess is not idempotent and thus requires *at least* two SCF iterations to ensure proper SCF convergence (idempotency of the density).

4.9.3 Core Hamiltonian

The core Hamiltonian guess [50] simply obtains the guess MO coefficients by diagonalizing the core Hamiltonian matrix (4.18). This approach works best with small basis sets.

4.9.4 GWH

The GWH guess procedure [47] uses a combination of the overlap matrix elements (4.12), and the diagonal elements of the Core Hamiltonian matrix (4.18) according to the relation

$$H_{\mu\upsilon} = c_x S_{\mu\upsilon} \left(H_{\mu\mu} + H_{\upsilon\upsilon} \right) / 2 \qquad (4.45)$$

where c_x is a constant.

4.9.5 Reading MOs from Disk

There are two methods by which MO coefficients can be used from a previous job by reading them from disk:

1. Running two independent jobs sequentially invoking *qchem* with three command line variables:

```
localhost-1> qchem job1.in job1.out save
localhost-2> qchem job2.in job2.out save
```

Notes: (1) The *$rem* variable *SCF_GUESS* must be set to *READ* in *job2.in.*
(2) Scratch files remain in *$QCSCRATCH/save* on exit.

2. Running a batch job where two jobs are placed into a single input file separated by the string “@@@” on a single line.

Notes: (1) The *$rem* variable *SCF_GUESS* must be set to *READ* in the second job of the batch file.
(2) A third qchem command line variable is not necessary.

(3) As for the SAD guess, Q-Chem requires at least two SCF cycles to ensure proper SCF convergence (idempotency of the density).

4.9.6 Features

- Superposition of Atomic Density (SAD)
- Core Hamiltonian
- Generalized Wolfsberg-Helmholtz Approximation
- Obtain the previous MOs from disk

4.9.7 Job Control

Uses may apply an alternative initial guess to the Q-Chem default by declaring the following *$rem* keyword variables in addition to requisite variables.

SCF_GUESS

Specifies the initial guess procedure to use for the SCF

VARIABLE:

STRING

DEFAULT:

SAD	Superposition of atomic density (available only with standard basis sets)

OPTIONS:

CORE	Diagonalize core Hamiltonian
SAD	Superposition of atomic density
GWH	Apply generalized Wolfsberg-Helmholtz approximation
READ	Read previous MOs from disk

RECOMMENDATION:

SAD guess, for standard basis sets. For general basis sets, use GWH guess and switch to Core Hamiltonian if problems are encountered.

4.9.8 Customization

It is sometimes useful for the occupied guess orbitals to be other than the lowest Nalpha (or Nbeta) orbitals. Q-Chem users may define the occupied guess orbitals using the *$occupied* keyword. Occupied guess orbitals are defined by listing the alpha orbitals to be occupied on the first line and beta on the second. Clearly virtual orbitals must exist in the initial guess which renders this option incompatible with the SAD guess.

```
$occupied
αi αj αk αl ... {alpha guess orbitals to be occupied}
βl βm βn βo ... {beta guess orbitals to be occupied}
$end
```

Figure 4.1 Format for modifying occupied guess orbitals.

SCF_GUESS_PRINT

Controls printing of guess MOs, Fock and density matrices

VARIABLE:

INTEGER

DEFAULT:

0 Do not print guesses

OPTIONS:

0 Do not print guesses

SAD

1 Atomic density matrics and molecular matrix

2 Level 1 plus density matrices

CORE and GWH

1 No extra output

2 Level 1 plus Fock and density matrices and, MO coefficients and eigenvalues

READ

1 No extra output

2 Level 1 plus density matrices, MO coefficients and eigenvalues

SCF_GUESS_MIX

Switch to control mixing of LUMO and HOMO to break symmetry in the initial guess.

VARIABLE:

LOGICAL

DEFAULT:

False Do not mix HOMO and LUMO in SCF guess

OPTIONS:

False Do not mix HOMO and LUMO in SCF guess

True Add 10% of LUMO to HOMO for SCF initial guess to break symmetry

SCF_GUESS_ALWAYS

Switch to force the regeneration of a new initial guess for each series of SCF iterations (for use in geometry optimization)

VARIABLE:

LOGICAL

DEFAULT:

False — Do not generate a new guess for each series of SCF iterations in an optimization; use MOs from the previous SCF calculation for the guess, if available

OPTIONS:

False — Do not generate a new guess for each series of SCF iterations in an optimization; use MOs from the previous SCF calculation for the guess, if available

True — Generate a new guess for each series of SCF iterations in a geometry optimization

4.10 Optimizing The SCF

4.10.1 Introduction

As for any numerical optimization procedure, the rate of convergence of the SCF procedure is dependent on the initial guess, and on the algorithm used to step towards the stationary point. Q-Chem features a number of alternative SCF optimization algorithms, including the highly successful DIIS procedures, a direct minimiser, a hybrid DIIS/direct minimizer scheme as well as the standard Roothaan repeated diagonalization.

4.10.2 DIIS

The SCF implementation of the Direct Inversion in the Iterative Subspace (DIIS) method [51, 52] uses the property of an SCF solution which requires the density matrix to commute with the Fock matrix

$$\mathbf{SPF} - \mathbf{FPS} = \mathbf{0} \tag{4.46}$$

During the SCF cycles, prior to achieving self-consistency, it is possible to define an error vector $\mathbf{e}_i$, which is non-zero

$$\mathbf{SP}_i\mathbf{F}_i - \mathbf{F}_i\mathbf{P}_i\mathbf{S} = \mathbf{e}_i \tag{4.47}$$

where $\mathbf{P}_i$ is obtained from diagonalization of $\hat{\mathbf{F}}_i$, and

$$\hat{\mathbf{F}}_k = \sum_{j=1}^{k-1} c_j \mathbf{F}_j \tag{4.48}$$

The DIIS coefficients c_k, are obtained by a least squares constrained minimisation of the error vectors, *viz*

$$Z = \left(\sum_k c_k \mathbf{e}_k \right) \cdot \left(\sum_k c_k \mathbf{e}_k \right) \tag{4.49}$$

where the constraint

$$\sum_k c_k = 1 \tag{4.50}$$

is imposed to yield a set of linear equations, of dimension (*N+1*)

$$\begin{pmatrix} \mathbf{e}_1 \cdot \mathbf{e}_1 & \cdots & \mathbf{e}_1 \cdot \mathbf{e}_N & 1 \\ \vdots & \ddots & \vdots & \vdots \\ \mathbf{e}_N \cdot \mathbf{e}_1 & \cdots & \mathbf{e}_N \cdot \mathbf{e}_N & 1 \\ 1 & \cdots & 1 & 0 \end{pmatrix} \begin{pmatrix} c_1 \\ \vdots \\ c_N \\ \lambda \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix} \tag{4.51}$$

Convergence criteria requires the largest element of the N^{th} error vector to be below a cutoff threshold, usually 10^{-5} for single point energies, often increased to 10^{-8} for optimizations and frequency calculations.

The rate of convergence may be improved by restricting the number of previous Fock matrices (size of the DIIS subspace, *$rem* variable DIIS_SUBSPACE_SIZE) used for determining the DIIS coefficients

$$\hat{\mathbf{F}}_k = \sum_{j=k-(L+1)}^{k-1} c_j \mathbf{F}_j \tag{4.52}$$

where *L* is the size of the DIIS subspace. As the Fock matrix nears self-consistency the linear matrix equations (4.51) tend to become severely ill-conditioned and it is often necessary to reset the DIIS subspace (this is automatically carried out by the program).

4.10.3 Direct Minimization

Q-Chem has implemented a highly efficient Direct Minimization scheme which also seeks a solution from an associated subspace. Q-Chem’s Direct minimizer tends to be more robust, but converges more slowly than the DIIS algorithm. Direct minimisation provides a useful alternative to DIIS.

4.10.4 Hybrid Direct/DIIS Minimisation

In a number of cases, the DIIS minimization procedure converges much more rapidly than the direct minimizer in early SCF cycles, but becomes inferior to direct minimization in later cycles as the linear DIIS equations become ill-conditioned. A hybrid scheme has, therefore, been implemented which uses the DIIS minimization procedure to achieve convergence to an intermediate cutoff threshold. Thereafter, the direct minimisation algorithm is used.

4.10.5 Features

- Pulay DIIS
- Direct minimizer
- Hybrid optimization procedure

4.10.6 Job Control

Convergence problems are sometimes encountered in quantum chemical procedures. Q-Chem users are able to modify the initial density matrix guess and the SCF algorithm in order to help alleviate any such problems. In general, users are able to modify the algorithm, convergence criteria and the maximum number of SCF cycles, but should also consider the precision used for ERI evaluation (*THRESH*).

SCF_ALGORITHM

Algorithm used for converging the SCF

VARIABLE:

STRING

DEFAULT:

DIIS	Pulay DIIS

OPTIONS:

DIIS	Pulay DIIS
DM	Direct minimizer
DIIS_DM	Uses DIIS initially, switching to direct minimizer for later iterations
ROOTHAAN	Roothaan repeated diagonalization

RECOMMENDATION:

Use DIIS unless wanting ROHF, in which case use the direct minimiser must be used

SCF_CONVERGENCE

SCF is considered converged when wavefunction error is less than $10^{-\mathrm{SCF\_CONVERGENCE}}$

VARIABLE:

INTEGER

DEFAULT:

5	for single point energy calculations
8	for geometry optimization and frequency calculations

OPTIONS:

User-defined

RECOMMENDATION:

Tighter criteria for geometry optimization and vibration analysis

MAX_SCF_CYCLES

Controls the maximum number of SCF iterations permitted

VARIABLE:

INTEGER

DEFAULT:

50

OPTIONS:

User-defined

4.10.7 CUSTOMIZATION

THRESH

Cutoff for neglect of two-electron integrals is $10^{-\text{THRESH}}$

VARIABLE:

INTEGER

DEFAULT:

8	for single point energies
10	for optimizations and frequency calculations

OPTIONS:

User-defined

RECOMMENDATION:

Should be at least 3 greater than *SCF_CONVERGENCE*

DIIS_SUBSPACE_SIZE

Controls the size of the DIIS subspace during the SCF

VARIABLE:

INTEGER

DEFAULT:

15

OPTIONS:

User-defined

DIIS_PRINT

Controls the output from DIIS SCF optimization

VARIABLE:

INTEGER

DEFAULT:

0

OPTIONS:

0	
1	Chosen method and DIIS coefficients & solutions
2	Level 1 plus changes in multipole moments
3	Level 2 plus Multipole moments
4	Level 3 plus extrapolated Fock matrices

4.10.8 EXAMPLE

```
$comment
ROHF/3-21G single point calculation on water cation
$end

$molecule
1 2
O
H1 O OH
H2 O OH H1 HOH

OH = 1.2
HOH = 120.0
$end

$rem
JOBTYPE             OPT       Optimise the geometry
EXCHANGE            HF        HF exchange
UNRESTRICTED        FALSE     ROHF
BASIS               3-21G     Basis set
SCF_ALGORITHM       DM        use the direct minimiser
SCF_CONVERGENCE     8         1.0d-08 convergence criteria
SCF_GUESS_ALWAYS    TRUE      new guess each OPT cycle
$end
```

Example 4.8 Q-Chem input for an ROHF optimization

4.11 MØLLER-PLESSET PERTURBATION THEORY

4.11.1 INTRODUCTION

The Hartree-Fock procedure, while often qualitatively correct, is frequently quantitatively deficient. The deficiency is due to the underlying assumption of the Hartree-Fock approximation: that electrons move *independently* within molecular orbitals subject to an averaged field imposed by the remaining electrons. The error that this introduces is called the correlation energy and a wide variety of procedures exist for estimating its magnitude. Because the accuracy of DFT-based correlation approaches is always limited by the quality of the functionals and grids used, it remains useful sometimes to turn to the more conventional approaches.

Møller-Plesset Perturbation Theory [54] is one of a series of conventional approaches of approximating the correlation energy of molecules. In particular, second order Møller-Plesset perturbation theory (MP2) is one of the simplest and most *useful* levels of theory beyond the Hartree-Fock approximation. Although the computational cost of MP2 is much greater than that of modern SCF procedures, it has been shown to be consistent andreliable in many application and to yield significantly improved energies, structures and frequencies [55] than the HF procedure.

4.11.2 Theoretical Background

The Hartree-Fock wave function (Ψ_0) and energy (E_0) are *approximate* solutions (eigenfunction and eigenvalue) to the exact Hamiltonian eigenvalue problem or Schrödinger's electronic wave equation (4.5). The HF wave function and energy are, however, exact solutions for the Hartree-Fock Hamiltonian (H_0) eigenvalue problem. If we assume that the Hartree-Fock wave function (Ψ_0) and energy (E_0) lie near the exact wave function (Ψ) and energy (E), we can now write the exact Hamiltonian operator as

$$H = H_o + \lambda V \tag{4.53}$$

where V is the small perturbation and λ is a dimensionless parameter. Expanding the exact wave function and energy in terms of the HF wave function and energy yields

$$E = E^{(0)} + \lambda E^{(1)} + \lambda^2 E^{(2)} + \lambda^3 E^{(3)} + \ldots \tag{4.54}$$

$$\Psi = \Psi_0 + \lambda \Psi^{(1)} + \lambda^2 \Psi^{(2)} + \lambda^3 \Psi^{(3)} + \ldots \tag{4.55}$$

substituting the expansions (4.53-55) into (4.5) and gathering terms in λ^n yields

$$H_0 \Psi_0 = E^{(0)} \Psi_0 \tag{4.56a}$$

$$H_0 \Psi^{(1)} + V\Psi_0 = E^{(0)} \Psi^{(1)} + E^{(1)} \Psi_0 \tag{4.56b}$$

$$H_0 \Psi^{(2)} + V\Psi^{(1)} = E^{(0)} \Psi^{(2)} + E^{(1)} \Psi^{(1)} + E^{(2)} \Psi_0 \tag{4.56c}$$

and so forth. Multiplying each of the equations (4.56) by Ψ_0 and integrating over all space yields the following expression for the n^{th} order (MPn) energy

$$E^{(0)} = \left\langle \Psi_0 \middle| H_0 \middle| \Psi_0 \right\rangle \tag{4.57a}$$

$$E^{(1)} = \left\langle \Psi_0 \middle| V \middle| \Psi_0 \right\rangle \tag{4.57b}$$

$$E^{(2)} = \left\langle \Psi_0 \middle| V \middle| \Psi^{(1)} \right\rangle \tag{4.57c}$$

Thus, the Hartree-Fock energy

$$E_0 = \left\langle \Psi_0 \middle| H_0 + V \middle| \Psi_0 \right\rangle \tag{4.58}$$

is simply the sum of the zeroth- and first- order energies

$$E_0 = E^{(0)} + E^{(1)} \tag{4.59}$$

The correlation energy can then be written

$$E_{corr} = E_0^{(2)} + E_0^{(3)} + E_0^{(4)} + \dots \tag{4.60}$$

of which the first term is the MP2 energy.

It can be shown [7, 50] that RHF MP2 energy can be written as

$$E_0^{(2)} = \frac{1}{4}\sum_{abrs}\frac{\left|\langle ab \| rs\rangle\right|^2}{\varepsilon_a + \varepsilon_b - \varepsilon_r - \varepsilon_s} \tag{4.61}$$

where

$$\langle ab \| rs\rangle = \langle ab | rs\rangle - \langle ab | sr\rangle \tag{4.62}$$

and

$$\langle ab | cd\rangle = \int \psi_a(\mathbf{r_1})\psi_c(\mathbf{r_1})\left[\frac{1}{r_{12}}\right]\psi_b(\mathbf{r_2})\psi_d(\mathbf{r_2})\,\mathrm{d}\,\mathbf{r_1}\,\mathrm{d}\,\mathbf{r_2} \tag{4.63}$$

which can be written in terms of the two electron repulsion integrals

$$\langle ab | cd\rangle = \sum_\mu\sum_\nu\sum_\lambda\sum_\sigma C_{\mu a}C_{\nu c}C_{\lambda b}C_{\sigma d}(\mu\nu|\lambda\sigma) \tag{4.64}$$

4.11.3 Q-Chem Implementation and Algorithms

Because MP2 offers a useful compromise between accuracy and cost, much research effort has been expended devising highly efficient programs for computing this correlation energy and its gradient. The implementation in Q-Chem is an improved version of that previously reported by Head-Gordon [56], particularly in the following areas:

- Uses pure functions, as opposed to Cartesians, for all fifth order steps. This leads to large computational savings for basis sets containing pure functions
- Customised loop unrolling for improved efficiency
- The sortless semi-direct method avoids a read and write operation resulting in a large I/O savings
- Reduction in disk and memory usage
- No extra integral evaluation for gradient calculations

The implementation offers the user three alternatives:

1. the program determines the optimal algorithm
2. direct algorithm
3. disk-based sortless semi-direct algorithm

The optimal algorithm is chosen on the basis of available memory. If sufficient memory is available to hold the two electron atomic orbital integrals and MP2 transformations in-core, then it will proceed on this basis. If there is insufficient memory available for the in-core algorithm, a check is made to determine if the algorithm can be carried out using the direct algorithm; otherwise the semi-direct approach is taken.

The semi-direct algorithm (energies only) determines if cubic or quadratic memory allocation is possible. If it is the latter, the algorithm optimizes the read block size on the basis of available ERI space.

4.11.4 FEATURES

- Direct
- Semi-direct

4.11.5 JOB CONTROL

Four *$rem* variables need to be set in order to run an MP2 calculation (*BASIS*, *JOBTYPE*, *EXCHANGE* and *CORRELATION*). *BASIS* is described in detail in Chapter 6, *JOBTYPE* defaults to single point, so this need only be set for any other type of job. The remaining variables are as follows:

EXCHANGE

Specifies the exchange level of theory.
OPTION:
HF Exact (Hartree-Fock)

CORRELATION

Specifies the correlation level of theory.
OPTION:
MP2 MP2

4.11.6 CUSTOMIZATION

The integral transformation algorithms used by Q-Chem (*e.g.*, MP2, CIS(D)) are limited by available disk space (D) and memory (C), the number of basis functions (N), the number of virtual orbitals (V) and the number of occupied orbitals (O).

MEMORY

Sets the memory for individual program modules
VARIABLE:
INTEGER
DEFAULT:
2,000,000 (2 MW)
OPTIONS:
User-defined number of words

MEMORY_TOTAL

Sets the total memory available to Q-Chem
VARIABLE:
INTEGER
DEFAULT:
Unlimited (1,000 MW)
OPTIONS:
User-defined number of words
RECOMMENDATION:
Use default

CD_MAX_DISK

Sets the amount of disk space (in words) available for MP2 calculations
VARIABLE:
INTEGER
DEFAULT:
60,000,000 (60 MW)
OPTIONS:
User-defined

CD_ALGORITHM

Determines the algorithm for MP2 integral transformations
VARIABLE:
STRING
DEFAULT:
Program determined
OPTIONS:

DIRECT	Uses fully direct algorithm
SEMI_DIRECT	Uses disk-based sortless semi-direct algorithm

RECOMMENDATION:
Use the default

N_FROZEN_CORE

Sets the number of frozen core orbitals in a post-Hartree-Fock calculation

VARIABLE:

INTEGER

DEFAULT:

0

OPTIONS:

FC Frozen Core approximation (all core orbitals frozen)

n Freeze *n* core orbitals

N_FROZEN_VIRTUAL

Sets the number of frozen virtual orbitals in a post-Hartree-Fock calculation

VARIABLE:

INTEGER

DEFAULT:

0

OPTIONS:

n Freeze *n* virtual orbitals

4.11.7 Examples

```
$molecule
0 1
O
H1 O OH
H2 O OH H1 HOH

OH = 1.01
HOH = 105
$end

$rem
JOBTYPE        SP       Single Point energy
CORRELATION    MP2
EXCHANGE       HF       Exact
BASIS          6-31G*
$end
```

Example 4.9 Example of an MP2/6-31G* calculation on the water molecule

```
$molecule
0 1
O
H1 O OH
H2 O OH H1 HOH

OH = 1.01
HOH = 105
$end

$rem
JOBTYPE         SP         Single Point energy
CORRELATION     MP2
EXCHANGE        HF         Exact
BASIS           6-31G*
N_FROZEN_CORE   FC         Frozen core approximation
$end
```

Example 4.10 Example of an MP2/6-31G* calculation employing the frozen core approximation

4.12 GROUND STATE METHOD SUMMARY

To summarise the main features of Q-Chem's ground state capabilities, the user needs to consider:

1. Input a molecular geometry (*$molecule* keyword)
   - Cartesian
   - *Z*-matrix
   - Read from prior calculations

2. Declare the job specification (*$rem* keyword)
   - JOBTYPE
     - ◊ Single point
     - ◊ Optimization
     - ◊ Frequency
   - BASIS
     - ◊ Refer to Chapter 6 (note: *$basis* keyword for user defined basis sets).
   - EXCHANGE
     - ◊ Linear scaling algorithms for all methods
     - ◊ Arsenal of exchange density functionals
     - ◊ User definable functionals and hybrids
   - CORRELATION
     - ◊ DFT or conventional methods
     - ◊ Linear scaling (CPU and memory) incorporation of correlation with DFT
     - ◊ Arsenal of correlation density functionals
     - ◊ User definable functionals and hybrids
     - ◊ Optimised MP2 implementation

3. Exploit Q-Chem's special features
   - QCTC/ONX, CFMM large molecule options
   - SCF rate of convergence increased through improved guessers, minimizers, incremental Fock matrix builds and variable integral thresholds
   - CASE approximation

4.13 References and Further Reading

Basis sets (Chapter 6)
Molecular Geometry Critical Points (Chapter 7)
Molecular Properties analysis (Chapter 8)
AOINTS (Appendix)

[1] E. Schrödinger, *Ann. Physik.* (1926), **79**, 361.

[2] W. Heisenberg, *Z. Physik.* (1926), **39**, 499.

[3] M. Born and J. R. Oppenheimer, *Ann. Physik.* (1927), **84**, 457.

[4] J. C. Slater, *Phys. Rev.* (1929), **34**, 1293.

[5] J. C. Slater, *Phys. Rev.* (1930), **35**, 509.

[6] J. A. Pople and R. K. Nesbet, *J. Chem. Phys.* (1954), **22**, 571.

[7] W. J. Hehre, L. Radom, P. v. R. Schleyer and J. A. Pople, *Ab initio molecular orbital theory*; Wiley: New York, 1986.

[8] R. D. Adamson, J. P. Dombroski and P. M. W. Gill, *Chem. Phys. Lett.* (1996), **254**, 329.

[9] J. P. Dombroski, S. W. Taylor and P. M. W. Gill, *J. Phys. Chem.* (1996), **100**, 6272.

[10] R. G. Parr and W. Yang, *Density-Functional theory of atoms and molecules*; First ed.; Oxford University Press: New York, 1989; Vol. 16.

[11] P. Hohenberg and W. Kohn, *Phys. Rev. B* (1964), **136**, 864.

[12] W. Kohn and L. J. Sham, *Phys. Rev. A* (1965), **140**, 1133.

[13] P. A. M. Dirac, *Proc. Cam. Phil. Soc.* (1930), **26**, 376.

[14] J. A. Pople, P. M. W. Gill and B. G. Johnson, *Chem. Phys. Lett.* (1992), **199**, 557.

[15] S. H. Vosko, L. Wilk and M. Nusair, *Can. J. Phys.* (1980), **58**, 1200.

[16] J. P. Perdew and A. Zunger, *Phys. Rev. B* (1981), **23**, 5048.

[17] E. P. Wigner, *Trans. Faraday Soc.* (1938), **34**, 678.

[18] A. D. Becke, *Phys. Rev. A* (1988), **38**, 3098.

[19] P. M. W. Gill, *Mol. Phys.* (1996), **89**, 443.

[20] C. Lee, W. Yang and R. G. Parr, *Phys. Rev. B* (1988), **37**, 785.

[21] J. P. Perdew, *Phys. Rev. B* (1986), **33**, 8822.

[22] J. R. Perdew, J. A. Chevary, S. H. Vosko, K. A. Jackson, M. R. Pederson, D. J. Singh and C. Fiolhais, *Phys. Rev. B* (1992), **46**, 6671.

[23] A. D. Becke, *J. Chem. Phys.* (1988), **88**, 2547.

[24] C. W. Murray, N. C. Handy and G. J. Laming, *Mol. Phys.* (1993), **78**, 997.

[25] Lebedev, *Sibirsk. Mat. Zh.* (1977), **18**, 132.

[26] Lebedev, *Zh. Vychisl. Mat. Mat. Fiz.* (1975), **15**, 48.

[27] Lebedev, *Zh. Vychisl. Mat. Mat. Fiz.* (1976), **16**, 293.

[28] P. M. W. Gill, B. G. Johnson and J. A. Pople, *Chem. Phys. Lett.* (1993), **209**, 506.

[29] B. G. Johnson, P. M. W. Gill and J. A. Pople, *Chem. Phys. Lett.* (1994), **220**, 377.

[30] A. D. Becke, *J. Chem. Phys.* (1993), **98**, 1372.

[31] R. D. Adamson, P. M. W. Gill and J. A. Pople, (in press).

[32] P. J. Stephens, F. J. Devlin, C. F. Chabalowski and M. J. Frisch, *J. Phys. Chem.* (1994), **98**, 11623.

[33] E. Schwegler and M. Challacombe, *J. Chem. Phys.* (1996), **106**, 9708.

[34] C. A. White and M. Head-Gordon, *J. Chem. Phys.* (1994), **101**, 6593.

[35] L. Greengard, *The rapid evaluation of potential fields in particle systems*; The MIT Press: London, 1987.

[36] C. A. White, B. G. Johnson, P. M. W. Gill and M. Head-Gordon, *Chem. Phys. Lett.* (1994), **230**, 8.

[37] C. A. White, B. G. Johnson, P. M. W. Gill and M. Head-Gordon, *Chem. Phys. Lett.* (1996), **253**, 268.

[38] C. A. White and M. Head-Gordon, *J. Chem. Phys.* (1996), **105**, 5061.

[39] C. A. White and M. Head-Gordon, *Chem. Phys. Lett.* (1996), **257**, 647.

[40] M. Challacombe, E. Schwegler and J. Almlöf, *J. Chem. Phys.* (1996), **104**, 4685.

[41] M. Challacombe and E. Schwegler, *J. Chem. Phys.* (1997), **106**, 5526.

[42] C. G. Lambert, T. A. Darden and J. A. Board, Jr., *J. Comput. Phys.* (1996), **126**, 274.

[43] R. Kutteh, E. Apra and J. Nichols, *Chem. Phys. Lett.* (1995), **238**, 173.

[44] M. C. Strain, G. E. Scuseria and M. J. Frisch, *Science* (1996), **271**, 51.

[45] H.-Q. Ding, N. Karasawa and W. A. Goddard III, *J. Chem. Phys.* (1992), **97**, 4309.

[46] T. Takada, M. Dupuis and H. F. King, *J. Chem. Phys.* (1981), **75**, 332.

[47] M. Dupuis and H. F. King, *Int. J. Quantum Chem.* (1977), **11**, 613.

[48] M. Dupuis and H. F. King, *J. Chem. Phys.* (1978), **68**, 3998.

[49] M. Wolfsberg and L. Helmholtz, *J. Chem. Phys.* (1952), **20**, 837.

[50] A. Szabo and N. S. Ostlund, *Modern quantum chemistry: Introduction to advanced electronic structure theory*; First (revised) ed.; McGraw-Hill: New York, 1989.

[51] P. Pulay, *J. Comput. Chem.* (1982), **3**, 556.

[52] P. Pulay, *Chem. Phys. Lett.* (1980), **73**, 393.

[53] C. Ochsenfeld and M. Head-Gordon, *Chem. Phys. Lett.* (1997), **270**, 399.

[54] C. Møller and M. S. Plesset, *Phys. Rev.* (1934), **46**, 618.

[55] D. J. DeFrees, B. A. Levi, S. K. Pollack, W. J. Hehre, J. S. Binkley and J. A. Pople, *J. Am. Chem. Soc* (1979), **101**, 4085.

[56] M. Head-Gordon, J. A. Pople and M. J. Frisch, *Chem. Phys. Lett.* (1988), **153**, 503.

[57] A. A. Jarecki and E. R. Davidson, *Chem. Phys. Lett.* (1999), **300**, 44.

CHAPTER 5 EXCITED STATE METHODS

5.1 GENERAL EXCITED STATE FEATURES

As for ground state calculations, performing an adequate excited state calculation involves making an appropriate choice of method and basis set. A hierarchy of single-reference *ab initio* methods is beginning to emerge for the treatment of excited states and the user must again strike a compromise between cost and accuracy.

In general, a basis set appropriate for a ground state density functional theory or Hartree-Fock calculations will be appropriate for describing valance excited states. However, many excited states involve significant contributions from very diffuse Rydberg orbitals, and, therefore, it is often advisable to use basis sets that include additional diffuse functions. The 6-31+G* basis set is a reasonable compromise for the low-lying valance excited states of many organic molecules, but to describe true Rydberg excited states, Q-Chem allows the user to add second and higher sets of diffuse functions (see Chapter 6).

Q-Chem supports three main types of excited state calculation:

- **Vertical absorption spectrum**
  This is the calculation of the excited states of the molecule at the ground state geometry, as appropriate for absorption spectroscopy. The methods supported for performing a vertical absorption calculation are: CIS, RPA, XCIS and CIS(D), each of which will be discussed in turn. In addition, it is possible to visualise the excited states either by attachment-detachment density analysis or by plotting the transition density. The visual analysis options are available only for the CIS method.

- **Excited state optimization**
  Optimization of the geometry of stationary points on excited state potential energy surfaces is valuable for understanding the geometric relaxation that occurs between the ground and excited state. Excited state optimization is currently available for UCIS and RCIS only.

- **Excited state vibrational analysis**
  Given an optimised excited state geometry, Q-Chem can calculate the force constants at the stationary point to predict excited state vibrational frequencies. Stationary points can also be characterized as minima, transition structures or n^{th}-order saddle points. Excited state vibrational analyses can only be performed using the UCIS and RCIS methods.

5.1.1 Attachment/Detachment Density Analysis

As methods for *ab initio* calculations of excited states are becoming increasingly more routine, the question is how best to extract chemical meaning from such calculations. Recently, a new method of analyzing molecular excited states has been proposed [1] which has proven very successful.

Consider the one-particle density matrices of the initial and final states of interest, $\mathbf{P}_1$ and $\mathbf{P}_2$ respectively. Assuming that each state is represented in a finite basis of spin-orbitals, such as the molecular orbital basis, and each state is at the same geometry. Subtracting these matrices yields the difference density

$$\Delta = \mathbf{P}_1 - \mathbf{P}_2 \tag{5.1}$$

Now, the eigenvectors of the one-particle density matrix **P** describing a single state are termed the natural orbitals, and provide the best orbital description that is possible for the state, in that a CI expansion using the natural orbitals as the single-particle basis is the most compact. The basis of the attachment/detachment analysis is to consider what could be termed natural orbitals of the electronic transition and their occupation numbers (associated eigenvalues). These are defined as the eigenvectors **U** defined by

$$\mathbf{U}^{\mathbf{t}}\Delta\mathbf{U} = \delta \tag{5.2}$$

The sum of the occupation numbers δ_p of these orbitals is then

$$tr(\Delta) = \sum_{p=1}^{N} \delta_p = n \tag{5.3}$$

where n is the net gain or loss of electrons in the transition. The net gain in an electronic transition which does not involve ionisation or electron attachment will obviously be zero.

The detachment density

$$\mathbf{D} = \mathbf{UdU}^{\mathbf{t}} \tag{5.4}$$

is defined as the sum of all natural orbitals of the difference density with negative occupation numbers, weighted by the absolute value of their occupations where **d** is a diagonal matrix with elements

$$d_p = -\min(\delta_p, 0) \tag{5.5}$$

The detachment density corresponds to the electron density associated with single particle levels vacated in an electronic transition or hole density.

The attachment density

$$\mathbf{A} = \mathbf{U}\mathbf{a}\mathbf{U}^{\mathbf{t}} \tag{5.6}$$

is defined as the sum of all natural orbitals of the difference density with positive occupation numbers where **a** is a diagonal matrix with elements

$$a_p = \max(\delta_p, 0) \tag{5.7}$$

The attachment density corresponds to the electron density associated with the single particle levels occupied in the transition or particle density. The difference between the attachment and detachment densities yields the original difference density matrix

$$\Delta = \mathbf{A} - \mathbf{D} \tag{5.8}$$

5.1.2 Relaxed/Unrelaxed CIS Densities

Properties associated with CIS excited states can be calculated using either an unrelaxed density, found from the square of the CIS wavefunction as in SCF theory, or, since the CIS energy does not obey the Hellman-Feynman theorem, by using a relaxed density. CIS fails to obey the Hellman-Feynman theorem because the molecular orbitals used have been optimized for the ground state, not the excited state.

The relaxed density is defined as the one-particle density occurring in CIS gradient theory, effectively compensates for the relaxation of the ground state molecular orbitals in the presence of an excitation. It has been found that CIS properties calculated with relaxed densities tend to be in better agreement with experiment, hence use of the relaxed density is preferred and recommended whenever possible.

Calculation of the relaxed density involves solving a single CPHF-like equation for each state. The cost of this procedure is approximately the same as the cost of the CIS calculation itself. The use of the CIS relaxed density is controlled by *$rem* logical variable *CIS_RELAXED_DENSITY*.

5.2 Non-Correlated Methods

Q-Chem includes several excited state methods which do not incorporate correlation: CIS, XCIS and RPA. These methods are sufficiently inexpensive that calculations on very large molecules are possible, and are roughly comparable to the HF treatment of the ground state in terms of performance. They tend to yield qualitative rather than quantitative insight.

5.2.1 CIS

The derivation of the CI-singles energy and wave function begins by selecting the HF single determinantal wave function as reference for the ground state of the system

$$\Psi_{HF} = \frac{1}{\sqrt{n!}} \det\left\{\chi_1 \chi_2 \cdots \chi_i \chi_j \cdots \chi_n\right\} \tag{5.9}$$

where *n* is the number of electrons, and the spin orbitals

$$\chi_i = \sum_{\mu}^{N} c_{\mu i} \phi_{\mu} \tag{5.10}$$

are expanded in a finite basis of *N* atomic basis functions. Molecular orbital coefficients $\{c_{\mu_i}\}$ are usually found by SCF procedures which solve the Hartree-Fock equations

$$\mathbf{FC} = \boldsymbol{\varepsilon}\mathbf{SC} \tag{5.11}$$

where **S** is the overlap matrix, **C** is the matrix of molecular orbital coefficients, **ε** is a diagonal matrix of orbital eigenvalues and **F** is the Fock matrix with elements

$$F_{\mu\upsilon} = H_{\mu\upsilon} + \sum_{\lambda\sigma} \sum_{i} c_{\mu i} c_{\upsilon i} \left(\mu\lambda \| \upsilon\sigma\right) \tag{5.12}$$

involving the core Hamiltonian and the antisymmetrized two-electron integrals

$$\left(\mu\nu \| \lambda\sigma\right) = \iint \phi_{\mu}(\mathbf{r}_1) \phi_{\nu}(\mathbf{r}_2) \left(1/r_{12}\right) \left[\phi_{\lambda}(\mathbf{r}_1) \phi_{\sigma}(\mathbf{r}_2) - \phi_{\lambda}(\mathbf{r}_2) \phi_{\sigma}(\mathbf{r}_1)\right] \mathrm{d}\,\mathbf{r}_1 \,\mathrm{d}\,\mathbf{r}_2 \tag{5.13}$$

On solving (5.11), the total energy of the ground state single determinant can be expressed as

$$E_{HF} = \sum_{\mu\upsilon} P_{\mu\upsilon}^{HF} H_{\mu\upsilon} + \frac{1}{2} \sum_{\mu\upsilon\lambda\sigma} P_{\mu\upsilon}^{HF} P_{\lambda\sigma}^{HF} \left(\mu\lambda \| \upsilon\sigma\right) + V_{nuc} \tag{5.14}$$

where P^{HF} is the HF density matrix and V_{nuc} is the nuclear repulsion energy.

Equation (5.9) represents only one of many possible determinants made from orbitals of the system; there are in fact *n*(*N* - *n*) possible singly substituted determinants constructed by replacing an orbital occupied in the ground state (*i*, *j*, *k*,...) with an orbital unoccupied in the ground state (*a*, *b*, *c*, ...). Such wave functions and energies can be written

$$\Psi_i^a = \frac{1}{\sqrt{n!}} \det\left\{\chi_1 \chi_2 \cdots \chi_a \chi_j \cdots \chi_n\right\} \tag{5.15}$$

$$E_{ia} = E_{HF} + \varepsilon_a - \varepsilon_i - \left(ia \| ia\right) \tag{5.16}$$

where we have introduced the antisymmetrized two-electron integrals in the molecular orbital basis

$$(pq||rs) = \sum_{\mu\upsilon\lambda\sigma} c_{\mu p} c_{\upsilon q} c_{\lambda r} c_{\sigma s} (\mu\lambda||\upsilon\sigma) \tag{5.17}$$

These singly excited wave functions and energies could be considered crude approximations to the excited states of the system. However, determinants of the form (5.15) are deficient in that they:

(1) do not yield pure spin states
(2) resemble more closely ionization rather than excitation
(3) are not appropriate for excitation into degenerate states

These deficiencies can be partially overcome by representing the exited state wavefunction as a linear combination of all possible singly excited determinants

$$\Psi_{CIS} = \sum_{ia} a_i^a \Psi_i^a \tag{5.18}$$

where the coefficients $\{a_{ia}\}$ can be obtained by diagonalizing the many-electron Hamiltonian, **A**, in the space of all single substitutions, where the matrix elements are

$$\begin{aligned} A_{ia,jb} &= \left\langle \Psi_i^a \middle| H \middle| \Psi_j^b \right\rangle \\ &= \left[E_{HF} + \varepsilon_a - \varepsilon_j\right]\delta_{ij}\delta_{ab} - (ja||ib) \end{aligned} \tag{5.19}$$

By Brillouin's theorem [2] single substitutions do not interact directly with a reference HF determinant, so the resulting eigenvectors from the CIS excited state represent a treatment roughly comparable to that of the HF ground state. The excitation energy is simply the difference between HF ground state energy and CIS excited state energies, and the eigenvectors of **A** correspond to the amplitudes of the single-electron promotions.

CIS calculations can be performed in Q-Chem using restricted (RCIS) [3, 4], unrestricted (UCIS), or restricted open shell (ROCIS) [5] spin orbitals.

5.2.2 RPA

The Random Phase Approximation (RPA) [6, 7] is an alternative to CIS for uncorrelated calculations of excited states. It offers some advantages for computing oscillator strengths, and is roughly comparable in accuracy to CIS for excitation energies to singlet states, but is inferior for triplet states. RPA energies are non-variational.

5.2.3 XCIS

The motivation for the extended CIS procedure (XCIS) [8] stems from the fact that ROCIS and UCIS are less effective for radicals that CIS is for closed shell molecules. Using the attachment-detachment density analysis procedure [1], the failing of ROCIS and UCIS methodologies for the nitromethyl radical [5] was traced to the neglect of a particular class of double substitution which involves the simultaneous promotion of an α spin electron from the singly occupied orbital and the promotion of a β spin electron into the singly occupied orbital. In particular, the spin adapted configurations

$$\left|\tilde{\Psi}_i^a(1)\right\rangle = \frac{1}{\sqrt{6}}\left(\Psi_{\bar{i}}^{\bar{a}} - \Psi_i^a\right) + \frac{2}{\sqrt{6}}\Psi_{p\bar{i}}^{a\bar{p}} \tag{5.20}$$

(where *a*, *b*, *c* ... are virtual orbitals, *i*, *j*, *k* ... are occupied orbitals and, *p*, *q*, *r* ... are singly occupied orbitals) are of crucial importance and , it is quite likely that similar excitations are also very significant in other radicals of interest.

The XCIS proposal, a more satisfactory generalization of CIS to open shell molecules, is to simultaneously include a restricted class of double substitutions similar to those in (5.20). To illustrate this, consider the resulting orbital spaces of an ROHF calculations: doubly occupied (*d*), singly occupied (*s*) and virtual (*v*). From this starting point we can distinguish three types of single excitations of the same multiplicity as the ground state: *d* → *s*, *s* → *v* and *d* → *v*. Thus, the spin adapted ROCIS wave function is

$$\left|\Psi_{ROCIS}\right\rangle = \frac{1}{\sqrt{2}}\sum_{ia}^{dv} a_i^a\left(\Psi_i^a + \Psi_{\bar{i}}^{\bar{a}}\right) + \sum_{pa}^{sv} a_p^a\Psi_p^a + \sum_{ip}^{ds} a_{\bar{i}}^{\bar{p}}\Psi_{\bar{i}}^{\bar{p}} \tag{5.21}$$

The extension of CIS theory to incorporate higher excitations maintains the ROHF as the ground state reference and adds terms to the ROCIS wave function similar to that of equation (5.21), as well as those where the double excitation occurs through different orbitals in the α and β space

$$\begin{aligned}\left|\Psi_{XCIS}\right\rangle = \frac{1}{\sqrt{2}}\sum_{ia}^{dv} a_i^a\left(\Psi_i^a + \Psi_{\bar{i}}^{\bar{a}}\right) + \sum_{pa}^{sv} a_p^a\Psi_p^a + \sum_{ip}^{ds} a_{\bar{i}}^{\bar{p}}\Psi_{\bar{i}}^{\bar{p}} + \\ \sum_{iap}^{dvs}\tilde{a}_i^a(p)\tilde{\Psi}_i^a(p) + \sum_{ia,p\neq q}^{dv,ss} a_{p\bar{i}}^{a\bar{q}}\Psi_{p\bar{i}}^{a\bar{q}}\end{aligned} \tag{5.22}$$

XCIS is defined only from a restricted open shell Hartree-Fock ground state reference, as it would be difficult to uniquely define singly occupied orbitals in a UHF wave function. In addition, β unoccupied orbitals, through which the spin-flip double excitation proceeds, may not match the half-occupied α orbitals in either character or even symmetry.

For molecules with closed shell ground states, both the HF ground and CIS excited states emerge from diagonalization of the Hamiltonian in the space of the HF reference and singly excited substituted configuration state functions. The XCIS case is different because the restricted class of double excitations included could mix with the ground state and lower its energy. This mixing is avoided to maintain the size consistency of the ground state energy.

With the inclusion of the restricted set of doubles excitations in the excited states, but not in the ground state, it could be expected that some fraction of the correlation energy be recovered, resulting in anomalously low excited state energies. However, the fraction of the total number of doubles excitations included in the XCIS wave function is very small and those introduced cannot account for the pair correlation of any pair of electrons. Thus, the XCIS procedure can be considered that which neglects electron correlation.

The computational cost of XCIS [8] is approximately four times greater than CIS and ROCIS, and its accuracy for open shell molecules is generally comparable to that of the CIS method for closed shell molecules. In general, it achieves qualitative agreement with experiment.

5.2.4 FEATURES

- Vertical absorption spectrum (CIS, RPA and XCIS.)
- Excited state optimization (RCIS and UCIS only)
- Excited state vibrational analysis (RCIS and UCIS only)
- Attachment/detachment density analysis (CIS only)
- Option to freeze virtual and core orbitals

5.2.5 JOB CONTROL

See also *BASIS*, *EXCHANGE* and *CORRELATION*. *EXCHANGE* must be *HF* and *CORRELATION* must be *None*.

CIS_N_ROOTS

Sets the number of CI-Singles (CIS) excited state roots to find
VARIABLE:
INTEGER
DEFAULT:

0	Do not look for any excited states

OPTIONS:

n > 0	Looks for *n* CIS excited states

CIS_SINGLES

Solve for singlet excited states in RCIS calculations (ignored for UCIS)
VARIABLE:
LOGICAL
DEFAULT:
True Solve for singlet states
OPTIONS:
True Solve for singlet states
False Do not solve for singlet states

CIS_TRIPLETS

Solve for triplet excited states in RCIS calculations (ignored for UCIS)
VARIABLE:
LOGICAL
DEFAULT:
True Solve for triplet states
OPTIONS:
True Solve for triplet states
False Do not solve for triplet states

RPA

Do an RPA calculation in addition to a CIS calculation
VARIABLE:
LOGICAL
DEFAULT:
False Do not do an RPA calculation
OPTIONS:
False Do not do an RPA calculation
True Do an RPA calculation

XCIS

Do an XCIS calculation in addition to a CIS calculation
VARIABLE:
LOGICAL
DEFAULT:
False Do not do an XCIS calculation
OPTIONS:
False Do not do an XCIS calculation
True Do an XCIS calculation (requires ROHF ground state)

JOBTYPE

Specifies the calculation.

VARIABLE:

STRING

DEFAULT:

SP	Single point energy

OPTIONS:

SP	Single point energy
OPT	Geometry Minimization
TS	Transition Structure Search
FREQ	Frequency Calculation
FORCE	Analytic Force calculation

RECOMMENDATION:

SP for vertical excitation energies
OPT for excited state optimizations
FREQ for vibrational analysis of excited states
See also *CIS_STATE_DERIV*.

CIS_STATE_DERIV

Sets CIS state for excited state optimizations and vibrational analysis

VARIABLE:

INTEGER

DEFAULT:

0	Does not select any of the excited states

OPTIONS:

n	Select the *n*th state

5.2.6 Customization

N_FROZEN_CORE

Controls the number of frozen core orbitals

VARIABLE:

INTEGER

DEFAULT:

0	No frozen core orbitals

OPTIONS:

FC	Frozen core approximation
n	Freeze *n* core orbitals

N_FROZEN_VIRTUAL

Controls the number of frozen virtual orbitals.
VARIABLE:
INTEGER
DEFAULT:
0 No frozen virtual orbitals
OPTIONS:
n Freeze *n* virtual orbitals

MAX_CIS_CYCLES

Maximum number of CIS iterative cycles allowed
VARIABLE:
INTEGER
DEFAULT:
30
OPTIONS:
User-defined
RECOMMENDATIONS:
Default is usually sufficient

CIS_CONVERGENCE

CIS is considered converged when error is less than $10^{-\text{CIS\_CONVERGENCE}}$
VARIABLE:
INTEGER
DEFAULT:
6 CIS convergence threshold 10^{-6}
OPTIONS:
User-defined

CIS_RELAXED_DENSITY

Use the relaxed CIS density for attachment/detachment density analysis
VARIABLE:
LOGICAL
DEFAULT:
False Do not use the relaxed CIS density in analysis
OPTIONS:
False Do not use the relaxed CIS density in analysis
True Use the relaxed CIS density in analysis

CIS_GUESS_DISK

Read the CIS guess from disk (previous calculation)
VARIABLE:
LOGICAL
DEFAULT:
False Create a new guess
OPTIONS:
False Create a new guess
True Read the guess from disk
RECOMMENDATIONS:
Requires a guess from previous calculation

CIS_GUESS_DISK_TYPE

Determines the type of guesses to be read from disk
VARIABLE:
INTEGER
DEFAULT:
Nil
OPTIONS:
0 Read triplets only
1 Read triplets and singlets
2 Read singlets only
RECOMMENDATIONS:
Must be specified if *CIS_GUESS_DISK* is *TRUE*

5.3 EXCITED STATE CORRELATED METHODS

5.3.1 CIS(D)

CIS(D) is a simple size-consistent doubles correction to CIS which has a computational cost which scales as the fifth power of the basis set for each excited state. In this sense, CIS(D) and can be considered an excited state analog of the ground state MP2 method. CIS(D) yields useful improvements in the accuracy of excitation energies relative to CIS, and yet can still be applied to relatively large molecules using Q-Chem's efficient integrals transformation package.

The CIS(D) excited state procedure [9, 10] is a second-order perturbative approximation to the computationally expensive CCSD, based on a single excitation configuration interaction (CIS) reference. The coupled cluster wavefunction, truncated at single and double excitations, is the exponential of the single and double substitution operators acting on the Hartree-Fock determinant

$$\left|\Psi\right\rangle = \exp\left(T_1 + T_2\right)\left|\Psi_0\right\rangle \tag{5.23}$$

Determining the singles and doubles amplitudes requires solving the two equations

$$\left\langle \Psi_i^a \middle| H - E \middle| \left(1 + T_1 + T_2 + \tfrac{1}{2}T_1^2 + T_1T_2 + \tfrac{1}{3!}T_1^3\right)\Psi_0 \right\rangle = 0 \tag{5.24}$$

$$\left\langle \Psi_{ij}^{ab} \middle| H - E \middle| \left(1 + T_1 + T_2 + \tfrac{1}{2}T_1^2 + T_1T_2 + \tfrac{1}{3!}T_1^3 + \tfrac{1}{2}T_2^2 + \tfrac{1}{2}T_1^2T_2 + \tfrac{1}{4!}T_1^4\right)\Psi_0 \right\rangle = 0 \tag{5.25}$$

which lead to the CCSD excited state equations, which can be written

$$\left\langle \Psi_i^a \middle| H - E \middle| \left(U_1 + U_2 + T_1U_1 + T_1U_2 + U_1T_2 + \tfrac{1}{2}T_1^2U_1\right)\Psi_0 \right\rangle = \omega b_i^a \tag{5.26}$$

$$\begin{aligned} \left\langle \Psi_i^a \middle| H - E \middle| \right. & \left(U_1 + U_2 + T_1U_1 + T_1U_2 + U_1T_2 + \tfrac{1}{2}T_1^2U_1 + T_2U_2 \right. \\ & \left. + \tfrac{1}{2}T_1^2U_2 + T_1T_2U_1 + \tfrac{1}{3!}T_1^3U_1 \middle| \Psi_0 \right\rangle = \omega b_{ij}^{ab} \end{aligned} \tag{5.27}$$

This is an eigenvalue equation $\mathbf{A}\boldsymbol{b} = \omega\boldsymbol{b}$ for the transition amplitudes ($\boldsymbol{b}$ vectors), which are also contained in the U operators.

The second-order approximation to the CCSD eigenvalue equation yields a second-order contribution to the excitation energy which can be written in the form

$$\omega^{(2)} = \boldsymbol{b}^{(0)\mathbf{t}}\mathbf{A}^{(1)}\boldsymbol{b}^{(1)} + \boldsymbol{b}^{(0)\mathbf{t}}\mathbf{A}^{(2)}\boldsymbol{b}^{(0)} \tag{5.28}$$

or in the alternative form

$$\begin{aligned} \omega^{(2)} &= \omega^{CIS(D)} \\ &= E^{CIS(D)} - E^{MP2} \end{aligned} \tag{5.29}$$

where

$$E^{CIS(D)} = \left\langle \Psi^{CIS} \middle| V \middle| U_2\Psi^{HF} \right\rangle + \left\langle \Psi^{CIS} \middle| V \middle| T_2U_1\Psi^{HF} \right\rangle \tag{5.30}$$

and

$$E^{MP2} = \left\langle \Psi^{HF} \middle| V \middle| T_2\Psi^{HF} \right\rangle \tag{5.31}$$

5.4 Q-Chem Implementation and Algorithms

5.4.1 CIS

Q-Chem now includes efficient analytical second derivatives of the CIS energy [11, 12], to yield excited state vibrational frequencies, force constants, polarizabilities, and infrared intensities. The semi-direct method used to evaluate the frequencies is generally similar to the semi-direct method used to evaluate Hartree-Fock frequencies for the ground state. Memory and disk requirements (see below) are similar, and the computer time scales approximately as the cube of the system size for large molecules.

While CIS excitation energies are relatively inaccurate, with errors of the order of 1eV, CIS excited state properties, such as frequencies, are much more useful. This is very similar to the manner in which ground state Hartree-Fock (HF) structures and frequencies are much more accurate than HF relative energies. Generally speaking, for low-lying excited states, it is expected that CIS vibrational frequencies will be systematically 10% higher or so relative to experiment [11-15]. If the excited states are of pure valence character, then basis set requirements are generally similar to the ground state. Excited states with partial Rydberg character require the addition of one or preferably two sets of diffuse functions.

The main complication associated with running analytical CIS second derivatives is ensuring Q-Chem has sufficient memory to perform the calculations. For most purposes, the defaults will be adequate, but if a large calculation fails due to a memory error, then the following additional information may be useful in fine tuning the input, and understanding why the job failed. Note that the analytical CIS second derivative code does not currently support frozen core or virtual orbitals (unlike Q-Chem's MP2 code). Unlike MP2 calculations, applying frozen core/virtual orbital approximations does not lead to large computational savings in CIS calculations as all computationally expensive steps are performed in the atomic basis.

The memory requirements for CIS (and HF) analytical frequencies are primarily extracted from "*C*" memory, which is defined as

$$\text{“}C\text{” memory} = MEMORY\_TOTAL - MEMORY \tag{5.32}$$

"*C*" memory must be large enough to contain a number of arrays whose size is 3**NAtoms***NBasis***NBasis* (*NAtoms* is the number of atoms and *NBasis* refers to the number of basis functions). The value of the *$rem* variable *MEMORY* should be set sufficiently large to permit efficient integral evaluation. If too large, it reduces the amount of "C" memory available. If too small, the job may fail due to insufficient scratch space. For most purposes, a value of about 5MW is sufficient and by default *MEMORY_TOTAL* is set to a very large number (large than physical memory on most computers) and thus *malloc* (memory allocation) errors may occur on jobs where the memory demands exceeds physical memory.

5.4.2 CIS(D)

Q-Chem's integral transformation package, which is also used for ground state MP2 calculations, contains several algorithms for four-index transformations of two-electron integrals. CIS(D) clearly requires such transformations and the user can choose from three alternatives:

1. program determines the optimal algorithm
2. direct algorithm
3. disk-based sortless semi-direct algorithm

The optimal algorithm is chosen on the basis of available memory. If sufficient memory is available to hold the two-electron atomic orbital integrals and CIS(D) (and MP2) transformations in-core, then the direct in-core algorithm is used. If there is insufficient memory available for the in-core algorithm a check is made to determine if the algorithm can be carried out using the direct algorithm; otherwise the semi-direct approach is taken. Refer to the section on the MP2 integral transform implementation in chapter 4.

A CIS(D) excited state calculation not only yields CIS(D) excitation energies, but also CIS excitation energies as well as HF and MP2 ground state energies.

5.4.3 FEATURES

- ROCIS(D), RCIS(D) and UCIS(D)
- Direct algorithm
- Semi-direct algorithm

5.4.4 JOB CONTROL

See also *BASIS* and *EXCHANGE $rem* variables. *EXCHANGE* must be *HF*.

CORRELATION

OPTION:

CIS(D)	CIS(D) excited states

CIS_N_ROOTS

Sets the number of CI-Singles (CIS) excited state roots to find.

VARIABLE:

INTEGER

DEFAULT:

0	Do not look for any excited states

OPTIONS:

n > 0	Look for *n* CIS excited states

5.4.5 CUSTOMIZATION

N_FROZEN_CORE

Controls the number of frozen core orbitals
VARIABLE:
INTEGER
DEFAULT:
0 No frozen core orbitals
OPTIONS:
FC Frozen core approximation
n Freeze *n* core orbitals

N_FROZEN_VIRTUAL

Controls the number of frozen virtual orbitals
VARIABLE:
INTEGER
DEFAULT:
0 No frozen virtual orbitals
OPTIONS:
n Freeze *n* virtual orbitals

CD_MAX_DISK

Sets the amount of disk space (in words) available for integral transforms
VARIABLE:
INTEGER
DEFAULT:
60,000,000 (60 MW)
OPTIONS:
User-defined

CD_ALGORITHM

Determines the algorithm for integral transformations
VARIABLE:
STRING
DEFAULT:
Program-determined
OPTIONS:
DIRECT Uses fully direct algorithm
SEMI_DIRECT Uses disk-based sortless semi-direct algorithm
RECOMMENDATION:
Use default

5.5 FURTHER READING

Basis sets (Chapter 6)
SCF (Chapter 4)

[1] M. Head-Gordon, A. A. Grana, D. Maurice and C. A. White, *J. Phys. Chem.* (1995), **99**, 14261.

[2] A. Szabo and N. S. Ostlund, *Modern quantum chemistry: Introduction to advanced electronic structure theory*; First (revised) ed.; McGraw-Hill: New York, 1989.

[3] J. E. Del Bene, R. Ditchfield and J. A. Pople, *J. Chem. Phys.* (1971), **55**, 2236.

[4] J. B. Foresman, M. Head-Gordon, J. A. Pople and M. J. Frisch, *J. Phys. Chem.* (1992), **96**, 135.

[5] D. Maurice and M. Head-Gordon, *Int. J. Quantum Chem.* (1995), **29**, 361.

[6] T. D. Bouman and A. E. Hansen, *Int. J. Quantum Chem. Sym* (1989), **23**, 381.

[7] A. E. Hansen, B. Voigt and S. Rettrup, *Int. J. Quantum Chem.* (1983), **23**, 595.

[8] D. Maurice and M. Head-Gordon, *J. Phys. Chem.* (1996), **100**, 6131.

[9] M. Head-Gordon, R. J. Rico, M. Oumi and T. J. Lee, *Chem. Phys. Lett.* (1994), **219**, 21.

[10] M. Head-Gordon, D. Maurice and M. Oumi, *Chem. Phys. Lett.* (1995), **246**, 114.

[11] D. Maurice and M. Head-Gordon, *Mol. Phys.* (1998), *accepted*.

[12] D. Maurice, PhD Thesis. " Single Electron Theories of Excited States" University of California: Berkeley, 1998.

[13] J. F. Stanton, J. Gauss, N. Ishikawa and M. Head-Gordon, *J. Chem. Phys.* (1995), **103**, 4160.

[14] S. Zilberg and Y. Haas, *J. Chem. Phys.* (1995), **103**, 20.

[15] C. M. Gittins, E. A. Rohlfing and C. M. Rohlfing, *J. Chem. Phys.* (1996), **105**, 7323.

CHAPTER 6 BASIS SETS

6.1 INTRODUCTION

A basis set is a set of functions combined linearly to model molecular orbitals (4.10). Basis functions can be considered as representing the atomic orbitals of the atoms and are introduced in quantum chemical calculations because the equations defining the molecular orbitals are otherwise very difficult to solve directly.

Many standard basis sets have been carefully optimised and tested over the years. In principle, a user would employ the largest basis set available in order to model molecular orbitals as accurately as possible. In practice, however, computational cost grows rapidly with the size of the basis set so a compromise must be sought between accuracy and cost. If this is systematically pursued, it leads to a "theoretical model chemistry" [1], that is, a well-defined energy procedure (*e.g.*, Hartree-Fock) in combination with a well defined basis set.

Basis sets have been constructed from Slater, Gaussian and delta functions. Slater functions were initially employed because they are considered "natural" and have the correct behaviour at the origin and asymptotically. However, the two electron repulsion integrals (ERIs) encountered when using Slater basis functions are expensive and difficult. Delta functions are used in several quantum chemistry programs. However, while codes incorporating delta functions are simple, thousands of functions are required to achieve accurate results, even for small molecules.

The most important basis sets are contracted sets of atom-centred Gaussian functions, where the numbers of basis functions used are related to the number of core and valence atomic orbitals, and whether the atom is light (H or He) or heavy (everything else). Contracted basis sets have been shown to be computationally efficient and to have the ability to yield chemical accuracy (see the Appendix on AOINTS). The Q-Chem program has been optimized to exploit basis sets of the contracted Gaussian function type and has a large number of built-in standard basis sets (developed by Dunning and Pople, among others) which the user can access quickly and easily.

The selection of a basis set for quantum chemical calculations is very important. It is sometimes possible to use small basis sets to obtain good chemical accuracy, but calculations can often be significantly improved by the addition of diffuse and polarisation functions. Consult the literature and reviews to aid your selection. Refer to the "Further Reading" section at the end of this chapter.

6.2 BUILT-IN BASIS SETS

Q-Chem is equipped with many standard basis sets. Q-Chem allows the user to identify the required basis set by its standard symbolic representation. The available built in basis sets are presented below. The four types are:

1. Pople basis sets
2. Old Dunning basis sets
3. New, correlation consistent Dunning basis sets
4. Ahlrichs basis sets

6.2.1 FEATURES

- Extra diffuse functions necessary for high quality excited state calculations
- Pople's standard basis sets
- Arsenal of Dunning derived basis sets
- Dunning's correlation consistent basis sets
- Ahlrichs basis sets
- Standard polarisation functions
- Basis sets are requested by symbolic representation
- *s*, *p*, *d*, *f* and *g* angular momentum types of basis functions
- Maximum number of shells per atom is 100
- Pure and Cartesian basis functions

6.2.2 JOB CONTROL

BASIS

Sets the basis set to be used

VARIABLE:

STRING

DEFAULT:

No default basis set

OPTIONS:

General, Gen	User-defined. See section below
Symbol	Use standard basis sets as in the table below

RECOMMENDATIONS:

Consult literature and reviews [2, 3] to aid your selection.

6.3 Basis Set Symbolic Representation

Examples are given in the tables below and follow the standard format generally adopted for specifying basis sets. The single exception applies to additional diffuse functions. These are best inserted in a similar manner to the polarisation functions; in parentheses with the light atom designation following heavy atom designation. (*i.e., heavy, light*). Use a period (.) as a place holder (see examples).

	STO-*j*(*k*+,*l*+)G(*m*,*n*)	*j*-21(*k*+,*l*+)G(*m*,*n*)	*j*-31(*k*+,*l*+)G(*m*,*n*)	*j*-311(*k*+,*l*+)G(*m*,*n*)		
j	2, 3, 6	3	4, 6	6		
k	# sets of heavy atom diffuse functions					
l	# sets of heavy atom diffuse functions					
m	d	2d	3d	df	2df	3df
n	p	2p	3p	pd	2pd	3pd

Table 6.1a Summary of Pople type basis sets available in the Q-Chem program

Symbolic name	Atoms Supported
STO-2G	H, He, Li→Ne, Na→Ar, K, Ca, Sr
STO-3G	H, He, Li→Ne, Na→Ar, K→Kr, Rb→Sb
STO-6G	H, He, Li→Ne, Na→Ar, K→Kr
3-21G	H, He, Li→Ne, Na→Ar, K→Kr, Rb→Xe, Cs
4-31G	H, He, Li→Ne, P→Cl
6-31G	H, He, Li→Ne, Na→Ar, K→Zn
6-311G	H, He, Li→Ne, Na→Ar, Ga→Kr

Table 6.1b Atoms supported for Pople basis sets available in Q-Chem (see the Table below for specific examples).

Basis set	**Atoms Supported**
3-21G	H, He, Li→Ne, Na→Ar, K→Kr, Rb→Xe, Cs
3-21+G	H, He, Na→Cl, Na→Ar
3-21G*	H, He, Na→Cl
6-31G	H, He, Li→Ne, Na→Ar, K→Zn
6-31+G	H, He, Li→Ne, Na→Ar
6-31G*	H, He, Li→Ne, Na→Ar, K→Zn
6-31G(d,p)	H, He, Li→Ne, Na→Ar, K→Zn
6-31G(.,+)G	H, He, Li→Ne, Na→Ar
6-31+G*	H, He, Li→Ne, Na→Ar
6-311G	H, He, Li→Ne, Na→Ar, Ga→Kr
6-311+G	H, He, Li→Ne, Na→Ar
6-311G*	H, He, Li→Ne Na→Ar, Ga→Kr
6-311G(d,p)	H, He, Li→Ne Na→Ar, Ga→Kr

Table 6.1c Examples of extended Pople basis sets

	SV(*k*+,*l*+)(*m*d,*n*p)	DZ(*k*+,*l*+)(*m*d,*n*p)	TZ(*k*+,*l*+)(*m*d,*n*p)
k	# sets of heavy atom diffuse functions		
l	# sets of heavy atom diffuse functions		
m	# sets of d functions on heavy atoms		
n	# sets of p functions on light atoms		

Table 6.2a Summary of Dunning-type basis sets available in the Q-Chem program

Symbolic name	**Atoms Supported**
SV	H, Li→Ne
DZ	H, Li→Ne, Al→Cl
TZ	H, Li→Ne

Table 6.2b Atoms supported for old Dunning basis sets available in Q-Chem

Basis set	**Atoms Supported**
SV	H, Li→Ne
SV*	H, B→Ne
SV(d,p)	H, B→Ne
DZ	H, Li→Ne, Al→Cl
DZ+	H, B→Ne
DZ++	H, B→Ne
DZ*	H, Li→Ne
DZ**	H, Li→Ne
DZ(d,p)	H, Li→Ne
TZ	H, Li→Ne
TZ+	H, Li→Ne
TZ++	H, Li→Ne
TZ*	H, Li→Ne
TZ**	H, Li→Ne
TZ(d,p)	H, Li→Ne

Table 6.2c Examples of extended Dunning basis sets

Basis Set	Basis Set
cc-pVDZ	aug-cc-pVDZ
cc-pVTZ	aug-cc-pVTZ
cc-pVQT	aug-cc-pVQT
cc-pCVDZ	aug-cc-pCVDZ
cc-pCVTZ	aug-cc-pCVTZ
cc-pCVQZ	aug-cc-pCVQZ

Table 6.3a Summary of Dunning’s correlation-consistent basis sets available in Q-Chem

Symbolic name	Atoms Supported
cc-pVDZ	H, He, B→Ne, Al→Ar
cc-pVTZ	H, He, B→Ne, Al→Ar
cc-pVQZ	H, He, B→Ne, Al→Ar
cc-pCVDZ	B→Ne
cc-pCVTZ	B→Ne
cc-pCVQZ	B→Ne
aug-cc-pVDZ	H, He, B→Ne, Al→Ar
aug-cc-pVTZ	H, He, B→Ne, Al→Ar
aug-cc-pVQZ	H, He, B→Ne, Al→Ar
aug-cc-pCVDZ	B→F
aug-cc-pCVTZ	B→Ne
aug-cc-pCVQZ	B→Ne

Table 6.3b Atoms supported Dunning correlation-consistent basis sets available in Q-Chem

Symbolic name	Atoms Supported
TZV	Li→Kr
VDZ	H→Kr
VTZ	H→Kr

Table 6.4 Atoms supported for Ahlrichs basis sets available in Q-Chem

6.3.1 CUSTOMIZATION

Q-Chem offers a number of standard and special customization features. One of the most important is that of supplying additional diffuse functions. Diffuse functions are often important for the purpose of studying anions and excited states of molecules. For the latter, it is often important to supply several additional diffuse functions. This can be achieved by splitting the standard basis set diffuse function set into multiple diffuse sets, using a special scaling factor (3.32) applied to the standard diffuse function exponent. This yields a geometric series of diffuse function exponents, starting with the original standard exponent value.

PRINT_GENERAL_BASIS

Controls print out of built in basis sets in input format

VARIABLE:

LOGICAL

DEFAULT:

False	Do not print out standard basis set information

OPTIONS:

TRUE	Print out standard basis set information
FALSE	Do not print out standard basis set information

RECOMMENDATIONS:

Useful for modification of standard basis sets

6.4 USER-DEFINED BASIS SETS ($BASIS)

6.4.1 INTRODUCTION

Users may, on occasion, prefer to use non-standard basis sets and, it is possible to declare user-defined basis sets in Q-Chem input (see Chapter 3 on Q-Chem inputs). The format for inserting a non-standard user-defined basis set is both logical and flexible and is described in detail in the job control section below. Note that the SAD guess is not currently supported with non-standard user-defined basis sets.

6.4.2 FEATURES

- Insertion of user-defined contracted basis sets
- Angular momentum types S, P, SP, D, F & G
- Pure or Cartesian functions
- Maximum number of shells per atom is 100

6.4.3 Job Control

BASIS

OPTION:

General, Gen User-defined basis *via $basis* keyword

PURECART

Controls the use of pure (spherical harmonic) or Cartesian angular forms

DEFAULT:

None.

OPTIONS:

gfd Use 1 for pure and 2 for Cartesian.

Must be defined for user supplied basis sets.

When using a non-standard basis set which incorporates *d* or higher basis functions of angular momentum, the *$rem* variable *PURECART* needs to be initiated. This *$rem* variable indicates to the Q-Chem program how to handle the angular form of the basis functions. As indicated above, each integer represents an angular momentum type which can be defined as either pure (1) or Cartesian (2). For example, 111 would indicate to the Q-Chem program to treat all G, F and D basis functions as being in the pure form. 121 would indicate G and D’s pure and F’s Cartesian, *etc*.

6.4.4 EXAMPLE

```
$molecule
0 1
O
H O OH
H O OH 2 HOH

OH = 1.2
HOH = 110.0
$end

$rem
EXCHANGE       HF        HF Exchange
BASIS          Gen       User-defined general basis
SCF_GUESS      GWH       SAD unavailable for general basis
PURECART       112       Cartesian D functions, Pure F and G
$end

$basis
H          0
S          1   1.00
           1.30976       0.430129
           0.233136      0.678914
****
O          0
S          2   1.00
           49.9810       0.430129
           8.89659       0.678914
SP         2   1.00
           1.94524       0.0494720      0.511541
           0.493363      0.963782       0.612820
D          1        1.00
           0.39     1.00
F          1        1.00
           4.1      1.00
G          1        1.00
           3.35     1.00
****
$end
```

Example 6.1 Example of adding a user-defined non-standard basis set. Note that since D, F and G functions are incorporated, the *$rem* variable *PURECART* must be set.

6.5 FORMAT FOR USER-DEFINED BASIS SETS

atomic_symbol	0			
ang_mom_sym		*contraction*_K	*scaling*	
*exp*_1	*coeff*_1_Lmin	*coeff*_1_(Lmin+1)	...	*coeff*_1_Lmax
*exp*_2	*coeff*_2_Lmin	*coeff*_2_(Lmin+1)	...	*coeff*_2_Lmax
*exp*_3	*coeff*_3_Lmin	*coeff*_2_(Lmin+1)	...	*coeff*_3_Lmax
.	.	.	.	.
.	.	.	.	.
*exp*_K	*coeff*_K_Lmin	*coeff*_K_(Lmin+1)	...	*coeff*_K_Lmax

atomic_symbol Atomic symbol of the atom (atomic number not accepted)
ang_mom_sym Angular momentum symbol (S, P, SP, D, F, G)
contraction Degree of contraction of the shell (integer)
scaling Scaling to be applied to exponents (default is 1.00)
exp Gaussian primitive exponent (positive real number)
coeff Contraction coefficient for each angular momentum (non-zero real numbers)

Atoms are terminated with **** and the complete basis set is terminated with the *$end* keyword terminator. No blank lines can be incorporated within the general basis set input. As with all Q-Chem input deck information, all input is case-insensitive.

6.5.1 CUSTOMIZATION

In addition to defining one's own basis set, it is possible to define separate standard basis sets for individual atoms or a combination of standard and non-standard basis sets, exclusively, for individual atoms. These can be entered by the more familiar symbolic representation.

Notes: (1) It is not possible to augment a standard basis set in this way; the whole basis needs to be inserted manually (angular momentum, exponents, contraction coefficients) and additional functions added. Standard basis set exponents and coefficients can be easily obtained by appropriately setting the *PRINT_GENERAL_BASIS $rem* variable to *TRUE*.
(2) The *PURECART* flag must be set for *all* general basis input containing D angular momentum or higher functions, regardless of whether standard basis sets are entered in this non-standard manner.

6.5.2 Examples

```
$molecule
0 1
O
H O OH
H O OH 2 HOH

OH = 1.2
HOH = 110.0
$end

$rem
EXCHANGE       HF        HF Exchange
BASIS          Gen       User-defined general basis
SCF_GUESS      GWH       SAD unavailable for general basis
$end

$basis
H          0
S          2   1.00
           1.30976       0.430129
           0.233136      0.678914
****
O          0
S          2   1.00
           49.9810       0.430129
           8.89659       0.678914
SP         2   1.00
           1.94524       0.0494720      0.511541
           0.493363      0.963782       0.612820
****
$end
```

Example 6.2 Example of adding a user-defined non-standard basis set.

```
$molecule
0 1
O
H O OH
H O OH 2 HOH

OH = 1.2
HOH = 110.0
$end

$rem
EXCHANGE       HF        HF Exchange
CORRELATION    None      No correlation energy
BASIS          General   User-defined general basis
SCF_GUESS      GWH       SAD unavailable for general basis
PURECART       1         Pure D functions
$end

$basis
H          0
S          2   1.00
           1.30976       0.430129
           0.233136      0.678914
****
O          0
S          2   1.00
           49.9810       0.430129
           8.89659       0.678914
SP         2   1.00
           1.94524       0.0494720      0.511541
           0.493363      0.963782       0.612820
D          1   1.00
           0.39          1.00000
****
$end
```

Example 6.3 Example of adding a user-defined non-standard basis set. Note that since D functions are incorporated the *$rem* variable *PURECART* has been initiated.

```
$molecule
0 1
O
H O OH
H O OH 2 HOH

OH = 1.2
HOH = 110.0
$end

$rem
EXCHANGE       HF        HF Exchange
BASIS          General   User-defined general basis
SCF_GUESS      GWH       SAD unavailable for general basis
$end

$basis
H          0
STO-2G
****
O          0
STO-6G
****
$end
```

Example 6.4 Example of adding a user-defined non-standard basis set where the user defines different standard basis sets for each atom.

```
$molecule
0 1
O
H O OH
H O OH 2 HOH

OH = 1.2
HOH = 110.0
$end

$rem
EXCHANGE       HF        HF Exchange
BASIS          General   User Defined general basis
PURECART       2         Cartesian D functions
SCF_GUESS      GWH       SAD unavailable for general basis
$end

$basis
H          0
6-31G
****
O          0
6-311G(d)
****
$end
```

Example 6.5 Example of adding a user defined non-standard basis set. The user is able to specify different standard basis sets for different atoms.

```
$molecule
0 1
O
H O OH
H O OH 2 HOH

OH = 1.2
HOH = 110.0
$end

$rem
EXCHANGE       HF        HF Exchange
BASIS          General   User-defined general basis
SCF_GUESS      GWH       SAD unavailable for general basis
$end

$basis
H          0
S          2   1.00
           1.30976       0.430129
           0.233136      0.678914
****
O          0
STO-6G
****
$end
```

Example 6.6 Example of adding a user-defined non-standard basis set. The user is able to specify standard basis sets for some atoms and supply user-defined exponents and contraction coefficients for others. This might be particularly useful in cases where the user has constructed exponents and contraction coefficients for atoms not defined in standard basis sets so that only the non-defined atoms need have the exponents and contraction coefficients entered.

6.6 FURTHER READING

[1] W. J. Hehre, L. Radom, P. v. R. Schleyer and J. A. Pople, *Ab initio molecular orbital theory*; Wiley: New York, 1986.

[2] S. Huzinaga, *Comp. Phys. Rep.* (1985), **2**, 279.

[3] E. R. Davidson and D. Feller, *Chem. Rev.* (1986), **86**, 681.

[4] Basis sets were obtained from the Extensible Computational Chemistry Environment Basis Set Database, Version 1.0, as developed and distributed by the Molecular Science Computing Facility, Environmental and Molecular Sciences Laboratory which is part of the Pacific Northwest Laboratory, P.O. Box 999, Richland, Washington 99352, USA, and funded by the U.S. Department of Energy. The Pacific Northwest Laboratory is a multi-program laboratory operated by Battelle Memorial Institute for the U.S. Department of Energy under contract DE-AC06-76RLO 1830. Contact David Feller, Karen Schuchardt or Don Jones for further information.

CHAPTER 7 MOLECULAR GEOMETRY CRITICAL POINTS

7.1 EQUILIBRIUM GEOMETRIES AND TRANSITION STRUCTURES

Molecular potential energy surfaces rely on the Born-Oppenheimer separation of nuclear and electronic motion. Minima on such energy surfaces correspond to the classical picture of equilibrium geometries and first-order saddle points to transition structures. Both equilibrium and transition structures are stationary points. Gradients of equilibrium and transition structures should vanish; characterisation of the critical point requires consideration of the eigenvalues of the Hessian (second derivative matrix). Equilibrium geometries have Hessians whose eigenvalues are all positive. Transition structures, on the other hand, have Hessians with exactly one negative eigenvalue. That is, a transition structure is a maximum along a reaction path between two local minima, but a minimum in all directions perpendicular to the path.

The quality of a geometry optimization algorithm is of major importance; even the fastest integral code in the world will be useless if combined with an inefficient optimization algorithm that requires excessive numbers of steps to converge. Thus, Q-Chem incorporates the most advanced geometry optimisation features currently available through Jon Baker's OPTIMIZE package (see Appendix), a product of over ten years of research and development.

The key to optimizing a molecular geometry successfully is to proceed from the starting geometry to the final geometry in as few steps as possible. Four factors influence the path and number of steps:

- starting geometry
- optimization algorithm
- quality of the Hessian (and gradient)
- coordinate system

Q-Chem controls the last three of these, but the starting geometry is solely determined by the user, and the closer it is to the converged geometry, the fewer optimization steps will be required. Decisions regarding the optimizing algorithm and the coordinate system are generally made by the OPTIMIZE package to maximise the rate of convergence. Users are able to override these decisions, but in general, this is not recommended.

Another consideration in minimising optimization time concerns gradient and Hessian quality. A higher quality Hessian (*i.e.*, analytical *vs.* approximate) will in many cases lead to faster convergence and hence, fewer optimization steps. However, the construction of an analytical Hessian requires significant computational effort and may outweigh the advantage of fewer optimization cycles. Currently available analytical gradients and Hessians are summarized in Table 7.1.

Level of Theory/ Algorithm	Analytical Gradients	Maximum Angular Momentum Type	Analytical Hessian	Maximum Angular Momentum Type
DFT	✓	g	✗	
HF	✓	g	✓	f
MP2	✓	g	✗	
CIS (except RO)	✓	g	✓	f
QCTC	✗		✗	
ONX	✗		✗	
CFMM	✓	g	✗	

Table 7.1 Gradients and Hessians currently available for geometry optimizations with maximum angular momentum types for analytical derivative calculations (for higher angular momentum, derivatives are computed numerically)

7.2 USER-CONTROLLABLE PARAMETERS

Note: Users input starting geometry through the *$molecule* keyword.

7.2.1 FEATURES

- Cartesian, *Z*-matrix or internal coordinate systems
- Eigenvector Following (EF) or GDIIS algorithms
- Constrained optimizations
- Equilibrium structure searches
- Transition structure searches
- Initial Hessian and Hessian update options

7.2.2 JOB CONTROL

Users must first define what level of theory is required. Refer back to previous sections regarding enhancements and customization of these features. *EXCHANGE, CORRELATION* (if required) and *BASIS $rem* variables must be set.

The remaining *$rem* variables are those specifically relating to the OPTIMIZE package.

JOBTYPE

Specifies the calculation

VARIABLE:

STRING

OPT	Equilibrium structure optimization
TS	Transition structure optimization

GEOM_OPT_HESSIAN

Hessian status

VARIABLE:

STRING

DEFAULT:

DIAGONAL

OPTIONS:

DIAGONAL	Set up (default) diagonal Hessian.
READ	Have exact or initial Hessian. Use as is if Cartesian, or transform if internals.

GEOM_OPT_COORDS

Controls the type of optimization coordinates

VARIABLE

INTEGER

DEFAULT:

-1	Generate and optimize in internal coordinates, if this fails at any stage of the optimization, switch to Cartesian and continue

OPTIONS:

0	Optimize in Cartesian coordinates
1	Generate and optimize in internal coordinates, if this fails abort
-1	Generate and optimize in internal coordinates, if this fails at any stage of the optimization, switch to Cartesian and continue
2	Optimize in *Z*-matrix coordinates, if this fails abort
-2	Optimize in *Z*-matrix coordinates, if this fails during any stage of the optimization switch to Cartesians and continue

RECOMMENDATION:

Use the default; delocalized internals are more efficient

GEOM_OPT_TOL_GRADIENT

Convergence on maximum gradient component

VARIABLE:

INTEGER

DEFAULT:

300 ≡ 300 x 10^{-6} tolerance on maximum gradient component

OPTIONS:

Integer value (tolerance = value x 10^{-6})

RECOMMENDATION:

Use the default. To converge *GEOM_OPT_TOL_GRADIENT* and one of *GEOM_OPT_TOL_DISPLACEMENT* and *GEOM_OPT_TOL_ENERGY* must be satisfied

GEOM_OPT_TOL_DISPLACEMENT

Convergence on maximum atomic displacement

VARIABLE:

INTEGER

DEFAULT:

1200 ≡ 1200 x 10^{-6} tolerance on maximum atomic displacement

OPTIONS:

Integer value (tolerance = value x 10^{-6})

RECOMMENDATION:

Use the default. To converge *GEOM_OPT_TOL_GRADIENT* and one of *GEOM_OPT_TOL_DISPLACEMENT* and *GEOM_OPT_TOL_ENERGY* must be satisfied

GEOM_OPT_TOL_ENERGY

Convergence on energy change of successive optimisation cycles

VARIABLE:

INTEGER

DEFAULT:

100 ≡ 100 x 10^{-8} tolerance on maximum gradient component

OPTIONS:

Integer value (tolerance = value x 10^{-8})

RECOMMENDATION:

Use the default. To converge *GEOM_OPT_TOL_GRADIENT* and one of *GEOM_OPT_TOL_DISPLACEMENT* and *GEOM_OPT_TOL_ENERGY* must be satisfied

GEOM_OPT_MAX_CYCLES

Maximum number of optimisation cycles
VARIABLE:
INTEGER
DEFAULT:
20
OPTIONS:
User defined positive integer
RECOMMENDATION:
Use the default, increase for difficult cases

GEOM_OPT_PRINT

Amount of OPTIMIZE print output
VARIABLE:
INTEGER
DEFAULT:

3	Error messages, summary, warning, standard information and gradient print out

OPTIONS:

0	Error messages only
1	Level 0 plus summary and warning print out
2	Level 1 plus standard information
3	Level 2 plus gradient print out
4	Level 3 plus hessian print out
5	Level 4 plus iterative print out
6	Level 5 plus internal generation print out
7	Debug print out

RECOMMENDATION:
Use the default

7.2.3 Customization

GEOM_OPT_SYMFLAG

Controls the use of symmetry in OPTIMIZE
VARIABLE:
INTEGER
DEFAULT:

1	Make use of point group symmetry

OPTIONS:

1	Make use of point group symmetry
0	Do not make use of point group symmetry

RECOMMENDATION:
Use default

GEOM_OPT_MODE

Determines Hessian mode followed during TS search

VARIABLE:

INTEGER

DEFAULT:

0	Mode following off

OPTIONS:

0	Mode following off
n	Maximise along mode *n*

RECOMMENDATION:

Use default

GEOM_OPT_MAX_DIIS

Controls maximum size of subspace for GDIIS

VARIABLE:

INTEGER

DEFAULT:

0	Do not use GDIIS

OPTIONS:

0	Do not use GDIIS
-1	Default size = min(NDEG, NATOMS, 4) NDEG = number of molecular degrees of freedom
n	Size specified by user

RECOMMENDATION:

Use default or do not set *n* too large

GEOM_OPT_DMAX

Maximum allowed step size. Value supplied is multiplied by 10^{-3}

VARIABLE:

INTEGER

DEFAULT:

300	= 0.3

OPTIONS:

n	User-defined cutoff

RECOMMENDATION:

Use default

GEOM_OPT_UPDATE

Controls the Hessian update algorithm

VARIABLE:

INTEGER

DEFAULT:

-1	Use the default update algorithm

OPTIONS:

-1	Use the default update algorithm
0	Do not update the Hessian (not recommended)
1	Murtagh-Sargent update
2	Powell update
3	Powell/Murtagh-Sargent update (TS default)
4	BFGS update (OPT default)
5	BFGS with safeguards to ensure retention of positive definiteness (GDISS default)

RECOMMENDATION:

Use default

GEOM_OPT_LINEAR_ANGLE

Threshold for near linear bond angles (degrees)

VARIABLE:

INTEGER

DEFAULT:

165 degrees

OPTIONS:

n	User-defined level

RECOMMENDATION:

Use default

Comment: Molecular Critical Points Beginning With Analytical Hessian

As outlined, the rate of convergence of the iterative optimization process is dependent on a number of factors, one of which is the use of an initial analytic Hessian. This is easily achieved by instructing Q-Chem to calculate an analytic Hessian and proceed then to determine the required critical point. This is illustrated in the following example.

7.2.4 Example

```
$molecule
0 1
O
H 1 OH
H 1 OH 2 HOH

OH = 1.1
HOH = 104
$end

$rem
JOBTYPE        FREQ           Calculate an analytic Hessian
EXCHANGE       HF
BASIS          6-31G(D)
$end

Now proceed with the Optimization making sure to read in the
analytic Hessian (use other available information too)

@@@
$molecule
READ
$end

$rem
JOBTYPE             OPT
EXCHANGE            HF
BASIS               6-31G(D)
SCF_GUESS           READ
GEOM_OPT_HESSIAN    READ      Have the initial Hessian
$end
```

Example 7.1 Geometry optimization for H_2O starting with an analytic Hessian

7.3 Constrained Optimization

7.3.1 Introduction

Constrained optimization refers to the optimization of molecular structures (transition or equilibrium) in which certain parameters (*e.g.*, bond lengths, bond angles or dihedral angles) are fixed. Jon Baker’s OPTIMIZE package implemented in the Q-Chem program has been modified to handle constraints directly in delocalized internal coordinates using the method of Lagrange multipliers (see appendix). Constraints are imposed in an *$opt* keyword section of the input file.

Features of constrained optimizations in Q-Chem are:

- Starting geometries do not have to satisfy imposed constraints
- Delocalized internal coordinates are the most efficient system for large molecules
- Q-Chem's free format *$opt* section allows the user to apply constraints with ease

Note: The *$opt* input section is case-insensitive and free-format, except that there should be no space at the start of each line.

7.3.2 GEOMETRY OPTIMIZATION WITH GENERAL CONSTRAINTS

CONSTRAINT and *ENDCONSTRAINT* define the beginning and end, respectively, of the constraint section of *$opt* within which users may specify up to six different types of constraints:

interatomic distances (angstroms, *value* > 0.0):

```
stre       atom1    atom2     value
```

angles (degrees, 180.0 ≥ *value* ≥ 0.0); *atom2* is the middle atom of the bend:

```
bend       atom1    atom2     atom3     value
```

out-of-plane-bends (degrees, 180.0 ≥ *value* ≥ -180.0); angle between *atom4* and the *atom1-atom2-atom3* plane:

```
outp       atom1    atom2     atom3     atom4     value
```

dihedral angles (degrees, 180.0 ≥ *value* ≥ -180.0); angle the plane *atom1-atom2-atom3* makes with the plane *atom2-atom3-atom4*:

```
tors       atom1    atom2     atom3     atom4     value
```

coplanar bends (degrees, 180.0 ≥ *value* ≥ -180.0); bending of *atom1-atom2-atom3* in the plane *atom2-atom3-atom4*:

```
linc       atom1    atom2     atom3     atom4     value
```

perpendicular bends (degrees, 180.0 ≥ *value* ≥ -180.0); bending of *atom1-atom2-atom3* perpendicular to the plane *atom2-atom3-atom4*:

```
linp       atom1    atom2     atom3     atom4     value
```

7.3.3 Frozen Atoms

Absolute atom positions can be frozen with the *FIXED* section. The section starts with the *FIXED* keyword as the first line and ends with the *ENDFIXED* keyword on the last. The format to fix a coordinate or coordinates of an atom is:

```
atom        coordinate_reference
```

coordinate_reference can be any combination of up to three characters *X*, *Y* and *Z* to specify the coordinate(s) to be fixed: *X*, *Y*, *Z*, *XY*, *XZ*, *YZ*, *XYZ*. The fixing characters must be next to each other. *e.g.*,

```
FIXED
2           XY
ENDFIXED
```

means the *x*-coordinate and *y*-coordinate of atom 2 are fixed, whereas

```
FIXED
2           X    Y
ENDFIXED
```

will yield erroneous results.

Note: When the *FIXED* section is specified within *$opt*, the optimization coordinates will be Cartesian.

7.3.4 Dummy Atoms

DUMMY defines the beginning of the dummy atom section and ENDDUMMY its conclusion. Dummy atoms are used to help define constraints during constrained optimizations in Cartesian coordinates. They cannot be used with delocalized internals.

All dummy atoms are defined with reference to a list of real atoms, that is, dummy atom coordinates are generated from the coordinates of the real atoms from the dummy atoms defining list (see below). There are three types of dummy atom:

1. Positioned at the arithmetic mean of the up to 7 real atoms in the defining list
2. Positioned a unit distance along the normal to a plane defined by three atoms, centred on the middle atom of the three
3. Positioned a unit distance along the bisector of a given angle

The format for declaring dummy atoms is:

```
DUMMY
idum       type     list_length    defining_list
ENDDUMMY
```

idum center number of defining atom (must be one greater than the total number of real atoms for the first dummy atom, two greater for second *etc*.)
type type of dummy atom (either 1, 2 or 3; see above)
list_length number of atoms in the defining list
defining_list list of up to 7 atoms defining the position of the dummy atom

Once defined, dummy atoms can be used to define standard internal (distance, angle) constraints as per the constraints section, above.

Warning: The use of dummy atoms of type 1 has never progressed beyond the experimental stage.

7.3.5 Dummy Atom Placement in Dihedral Constraints

Bond and dihedral angles cannot be constrained in Cartesian optimizations to exactly 0° or ±180°. This is because the corresponding constraint normals are zero vectors. Also, dihedral constraints near these two limiting values (within, say 20°) tend to oscillate and are difficult to converge.

These difficulties can be overcome by defining dummy atoms and redefining the constraints with respect to the dummy atoms. For example, a dihedral constraint of 180° can be redefined to two constraints of 90° with respect to a suitably positioned dummy atom. The same thing can be done with a 180° bond angle (long a familiar use in *Z*-matrix construction).

Typical usage is as follows:

Internal Coordinates	**Cartesian Coordinates**

`$opt`
`CONSTRAINT`
`tors I J K L 180.0`
`ENDCONSTRAINT`
`$end`
	`$opt`
	`DUMMY`
	`M 2 I J K`
	`ENDDUMMY`
	`CONSTRAINT`
	`tors I J K M 90`
	`tors M J K L 90`
	`ENDCONSTRAINT`
	`$end`

Table 7.2 Comparison of dihedral angle constraint method for adopted coordinates.

The order of atoms is important to obtain the correct signature on the dihedral angles. For a 0° dihedral constraint, J and K should be switched in the definition of the second torsion constraint in Cartesian coordinates.

Note: In almost all cases the above discussion is somewhat academic, as internal coordinates are now best imposed using delocalized internal coordinates and there is no restriction on the constraint values.

7.3.6 ADDITIONAL ATOM CONNECTIVITY

Normally delocalized internal coordinates are generated automatically from the input Cartesian coordinates. This is accomplished by first determining the atomic connectivity list (*i.e.*, which atoms are formally bonded) and then constructing a set of individual primitive internal coordinates comprising all bond stretches, all planar bends and all proper torsions that can be generated based on the atomic connectivity. The delocalized internal are in turn constructed from this set of primitives.

The atomic connectivity depends simply on distance and there are default bond lengths between all pairs of atoms in the code. In order for delocalized internals to be generated successfully, all atoms in the molecule must be formally bonded so as to form a closed system. In molecular complexes with long, weak bonds or in certain transition states where parts of the molecule are rearranging or dissociating, distances between atoms may be too great for the atoms to be regarded as formally bonded, and the standard atomic connectivity will separate the system into two or more distinct parts. In this event, the generation of delocalized internal will fail.

Additional atomic connectivity can be included for the system to overcome this difficulty.

CONNECT defines the beginning of the additional connectivity section and *ENDCONNECT* the end. The format of the *CONNECT* section is:

```
CONNECT
atom      list_length   list
ENDCONNECT
```

atom	atom for which additional connectivity is being defined
list_length	number of atoms in the list of bonded atoms
list	list of up to 8 atoms considered as being bonded to the given atom

7.3.7 Example

```
$comment
methanol geom opt with constraints in bond length and bond
angles.
$end

$molecule
0 1
C      0.141915            0.332682        0.000000
O      0.141915           -1.088318        0.000000
H      1.186989            0.656186        0.000000
H     -0.348433            0.742676        0.887862
H     -0.348433            0.742676       -0.887862
H     -0.773953           -1.385902        0.000000
$end

$rem
GEOM_OPT_PRINT      6
GEOM_OPT_COORDS     2
NBO                 OFF
JOBTYPE             OPT
EXCHANGE            HF
BASIS               3-21G     Basis Set
$end

$opt
CONSTRAINT
stre 1 6 1.8
bend 2 1 4 110.0
bend 2 1 5 110.0
ENDCONSTRAINT
$end
```

Example 7.2 Methanol geometry optimization with constraints in posed on bond lengths and angles.

7.3.8 Summary

```
$opt
CONSTRAINT
stre       atom1    atom2     value
...
bend       atom1    atom2     atom3     value
...
outp       atom1    atom2     atom3     atom4     value
...
tors       atom1    atom2     atom3     atom4     value
...
linc       atom1    atom2     atom3     atom4     value
...
linp       atom1    atom2     atom3     atom4     value
...
ENDCONSTRAINT
FIXED
atom       coordinate_reference
...
ENDFIXED
DUMMY
idum       type     list_length    defining_list
...
ENDDUMMY
CONNECT
atom       list_length   list
...
ENDCONNECT
$end
```

7.4 Further Reading

Appendix: Geometry optimization with Q-Chem.

CHAPTER 8 MOLECULAR PROPERTIES AND ANALYSIS

8.1 INTRODUCTION

Q-Chem has incorporated a number of molecular properties and wavefunction analysis tools, summarised as follows:

- Chemical solvent models
- Population analysis
- Vibrational analysis
- Interface to the Natural Bond Orbital package
- Molecular orbital symmetries
- Multipole moments

8.2 CHEMICAL SOLVENT MODELS

Ab initio quantum chemical programs enable the accurate study of large molecules properties in the gas phase. However, some of these properties change significantly in solution. The largest changes are expected when going from vapour to polar solutions. Although in principle it is possible to model solvation effects upon the solute properties by supermolecular (cluster) calculations (*e.g.*, by averaging over several possible configurations of the first solvation shell)., these calculations are very demanding. Furthermore, the supermolecular calculations cannot, at present, provide accurate and stable *hydration energies*, for which long-range effects are very important. An accurate prediction of the hydration free energies is necessary for computer modelling of chemical reactions and ligand-receptor interactions in aqueous solution.

8.2.1 ONSAGER DIPOLE CONTINUUM SOLVENT

Q-Chem offers a solvent model based on that originally attributed to Onsager [1] in which the solute is placed in a spherical cavity surrounded by a continuous medium. The Onsager model requires two parameters: the cavity radius a_0 and a dielectric constant ε. Typically, the cavity radius is calculated using

$$a_0^3 = 3V_m/4\pi N \tag{8.1}$$

where V_m is obtained from experiment (molecular weight/density) [2] and N is Avogadro's number. It is also common to add 0.5 Å to the value of a_0 from (8.1) to account for the first solvation shell [3].

See the review by Tomasi and Perisco [4] for further insights into continuum solvent models.

8.2.2 JOB CONTROL

In order to run a calculation incorporating the Onsager solvent model, Q-Chem requires at least single point energy calculation *$rem* variables *BASIS*, *EXCHANGE* and *CORRELATION* (if required) in addition to the Onsager specific variables *SOLUTE_RADIUS* and *SOLVENT_DIELECTRIC*.

SOLUTE_RADIUS

Sets the Onsager solvent model cavity radius

VARIABLE:

INTEGER a_0 = SOLUTE_RADIUS/10000

DEFAULT:

No default

OPTIONS:

User-defined

RECOMMENDATION:

Use equation (8.1)

SOLVENT_DIELECTRIC

Sets the dielectric constant of the Onsager solvent continuum

VARIABLE:

INTEGER ε = SOLVENT_DIELECTRIC/10000

DEFAULT:

No default

OPTIONS:

User-defined

RECOMMENDATION:

As per required solvent

8.3 WAVEFUNCTION ANALYSIS

Q-Chem performs a number of standard wavefunction analyses by default. Switching the *$rem* variable *WAVEFUNCTION_ANALYSIS* to *FALSE* will prevent the calculation of all wavefunction analysis features (described in this section). Alternatively, each wavefunction analysis feature may be controlled by its *$rem* variable. (The NBO package which is interfaced with Q-Chem is capable of performing more sophisticated analyses. See later in this chapter and the NBO manual for details).

WAVEFUNCTION_ANALYSIS

Controls the running of the default wavefunction analysis tasks

VARIABLE:

LOGICAL

DEFAULT:

TRUE	Perform default wavefunction analysis

OPTIONS:

TRUE	Perform default wavefunction analysis
FALSE	Do not perform default wavefunction analysis

Note: *WAVEFUNCTION_ANALYSIS* has no effect on NBO, solvent models or vibrational analyses.

8.3.1 Population Analysis

The one-electron charge density, usually written as

$$\rho(\mathbf{r}) = \sum_{\mu\nu} P_{\mu\nu}\phi_{\mu}(\mathbf{r})\phi_{\nu}(\mathbf{r}) \tag{8.2}$$

represents the probability of finding an electron at the point **r**, but implies little regarding the number of electrons associated with a given nucleus in a molecule. However, since the number of electrons N is related to the occupied orbitals ψ_i by

$$N = 2\sum_{a}^{N/2} \left|\psi_a(\mathbf{r})\right|^2 \mathrm{d}\,\mathbf{r} \tag{8.3}$$

we can substitute the basis expansion of ψ_a into (8.3) and obtain

$$N = \sum_{\mu}\sum_{\upsilon} P_{\mu\upsilon}S_{\mu\upsilon} = \sum_{\mu}(\mathbf{PS})_{\mu\mu} = tr\mathbf{PS} \tag{8.4}$$

where we interpret $(\mathbf{PS})_{\mu\mu}$ as the number of electrons associated with ϕ_μ. If the basis functions are atom-centred, the number of electrons associated with a given atom can be obtained by summing over all the basis functions. This leads to the Mulliken formula for the net charge of the atom A

$$q_A = Z_A - \sum_{\mu\in A}(\mathbf{PS})_{\mu\mu} \tag{8.5}$$

where Z_A is the atom's nuclear charge. This is called a Mulliken population analysis [5]. Q-Chem performs a Mulliken population analysis by default.

POP_MULLIKEN

Controls running of Mulliken population analysis
VARIABLE:
LOGICAL
DEFAULT:
TRUE Calculate Mulliken populations
OPTIONS:
TRUE Calculate Mulliken populations
FALSE Do not calculate Mulliken populations
RECOMMENDATION:
TRUE Trivial additional calculation

8.3.2 Multipole Moments

Q-Chem can compute Cartesian multipole moments of the charge density to arbitrary order.

MULTIPOLE_ORDER

Determines highest order of multipole moments to print if wavefunction analysis requested
VARIABLE:
INTEGER
DEFAULT:
4
OPTIONS:
n Calculate moments to n^{th} order

8.3.3 Symmetry Decomposition

Q-Chem's default is to write the SCF wave function molecular orbital symmetries and energies to the output file. If requested, a symmetry decomposition of the kinetic and nuclear attraction energies can also be calculated.

SYMMETRY_DECOMPOSITION

Determines symmetry decompositions to calculate
VARIABLE:
INTEGER
DEFAULT:
1 Calculate MO eigenvalues and symmetry (if available)
OPTIONS:
0 No symmetry decomposition
1 Calculate MO eigenvalues and symmetry (if available)
2 Perform symmetry decomposition of kinetic energy and nuclear attraction matrices

8.4 Vibrational Analysis

Vibrational analysis is an extremely important tool for the quantum chemist, supplying a molecular fingerprint which is invaluable for aiding identification of molecular species in many experimental studies. Q-Chem includes a vibrational analysis package that can calculate vibrational frequencies and their Raman [6] and infrared activities. Vibrational frequencies are calculated by either using an analytic Hessian (if available, Table 7.1) or, numerical finite difference of the gradient. The default setting in Q-Chem is to use the highest analytical derivative order available for the requested theoretical method.

Following a vibrational analysis, Q-Chem computes useful statistical thermodynamic properties at standard temperature and pressure, including: zero-point vibration energy (ZPVE) and, translational, rotational and vibrational, entropies and enthalpies.

The performance of various *ab initio* theories in determining vibrational frequencies has been well documented. See references [7-9].

8.4.1 Job Control

In order to carry out a frequency analysis users must *at a minimum* provide a molecule within the *$molecule* keyword and define an appropriate level of theory within the *$rem* keyword using the *$rem* variables *EXCHANGE*, *CORRELATION* (if required) (Chapter 4) and *BASIS* (Chapter 6). Since the default type of job (*JOBTYPE*) is a single point energy (*SP*) calculation, the *JOBTYPE $rem* variable must be set to *FREQ*.

JOBTYPE

Specifies the calculation.
VARIABLE:
STRING
OPTION:

FREQ	Frequency Calculation

8.4.2 Customization

The standard output from a frequency analysis is extensive and includes:

- Vibrational frequencies
- Raman and IR activities and intensities (requires *$rem DORAMAN*)
- Atomic masses
- Zero-point vibrational energy
- Translational, rotational, and vibrational, entropies and enthalpies

DORAMAN

Controls calculation of Raman intensities. Requires *JOBTYPE* to be set to *FREQ*

VARIABLE:

LOGICAL

DEFAULT:

FALSE	Do not calculate Raman intensities

OPTIONS:

FALSE	Do not calculate Raman intensities
TRUE	Do calculate Raman intensities

VIBMAN_PRINT

Controls level of extra print out for vibrational analysis

VARIABLE:

INTEGER

DEFAULT:

1	Standard full information print out

OPTIONS:

1	Standard full information print out
3	Level 1 plus vibrational frequencies in atomic units
4	Level 3 plus mass-weighted Hessian matrix, projected mass-weighted Hessian matrix
6	Level 4 plus vectors for translations and rotations projection matrix

RECOMMENDATION:

Use default

8.5 Interface to the NBO Package

Q-Chem has incorporated the Natural Bond Orbital package (v4.0) for molecular properties and wavefunction analysis. The NBO package is invoked either by setting the *$rem* variable NBO to TRUE and is initiated after the SCF wavefunction is obtained. Users are referred to the NBO users manual for options and details relating to exploitation of the features offered in this package.

8.5.1 JOB CONTROL

If switched on for a geometry optimization, the NBO package will only be invoked at the end of the last optimization step.

NBO

Controls the use of the NBO package

VARIABLE:

LOGICAL

DEFAULT:

FALSE	Do not invoke the NBO package

OPTIONS:

FALSE	Do not invoke the NBO package
TRUE	Do invoke the NBO package

```
$nbo
{NBO program keywords, parameters and options}
$end
```

Figure 8.1 General format for requesting the NBO program from Q-Chem.

Notes: (1) *$rem* variable *NBO* must be set to *TRUE* before the *$nbo* keyword is recognized.
(2) Q-Chem does not currently support facets of the NBO package which require multiple job runs.

8.6 FURTHER READING

Ground State Methods (Chapter 4)
Basis Sets (Chapter 6)
NBO manual

[1] L. Onsager, *J. Am. Chem. Soc* (1936), **58**, 1486.

[2] R. C. Weast, *Handbook of chemistry and physics*; 70 ed.; R. C. Weast, Ed.; CRC Press: Boca Raton, FL, 1989.

[3] M. W. Wong, M. J. Frisch and K. B. Wiberg, *J. Am. Chem. Soc* (1991), **113**, 4776.

[4] J. Tomasi and M. Perisco, *Chem. Rev.* (1994), **94**, 2027.

[5] A. Szabo and N. S. Ostlund, *Modern quantum chemistry: Introduction to advanced electronic structure theory*; First (revised) ed.; McGraw-Hill: New York, 1989.

[6] B. G. Johnson and J. Florián, *Chem. Phys. Lett.* (1995), **247**, 120.

[7] C. W. Murray, G. J. Laming, N. C. Handy and R. D. Amos, *Chem. Phys. Lett.* (1992), **199**, 551.

[8] A. P. Scott and L. Radom, *J. Phys. Chem.* (1996), **100**, 16502.

[9] B. G. Johnson, P. M. W. Gill and J. A. Pople, *J. Chem. Phys.* (1993), **98**, 5612.

CHAPTER 9 EXTENDED CUSTOMIZATION

9.1 USER-DEPENDENT AND MACHINE-DEPENDENT CUSTOMIZATION

Q-Chem has developed a simple mechanism for users to set user-defined long-term defaults to override the built-in program defaults. Such defaults may be most suited to machine specific features such as memory allocation, as the total available memory will vary from machine to machine depending on specific hardware and accounting configurations. However, users may identify other important uses for this customization feature.

Q-Chem obtains input initialization variables from four sources:

1. user input file
2. *$HOME/.qchemrc* file
3. *$QC/config/preferences* file
4. program defaults

The order of preference of initialisation is summarised in the diagram contained in Figure 9.1, where the higher placed input mechanism overrides the lower.

Details of the requirements of the Q-Chem input file have been discussed in detail in this manual and in addition, many of the various program defaults which have been set by Q-Chem. However, in reviewing the variables and defaults, users may identify *$rem* variable defaults that they find too limiting or, variables which they find repeatedly need to be set within their input files for maximum exploitation of Q-Chem's features. Rather than continually having to remember to place such variables into the Q-Chem input file, users are able to set long-term defaults which are read each time the user runs a Q-Chem job. This is done by placing these defaults into the file *.qchemrc* stored in the users home directory. Additionally, system administrators can override Q-Chem defaults with an additional *preferences* file in the *$QC/config* directory achieving a hierarchy of input as illustrated in figure 9.1.

Note: The *.qchemrc* and *preferences* files are not requisites for running Q-Chem and currently only support *$rem* keywords.

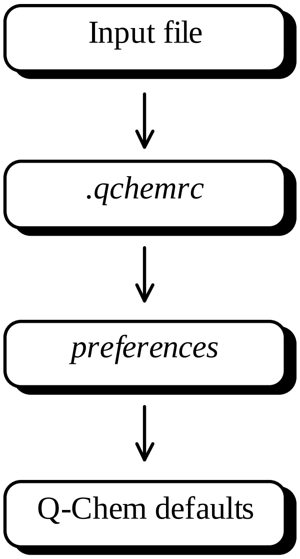

Figure 9.1 Diagram of input initialisation override settings. The higher mechanism indicates override preference of lower mechanisms of input.

9.1.1 .QCHEMRC AND PREFERENCES FILE FORMAT

The format of the *.qchemrc* and *preferences* files is similar to that for the input file, except that only a *$rem* keyword section may be entered, terminated with the usual *$end* keyword. Any other keyword sections will be ignored.

It is important that the *.qchemrc* and *preferences* files have appropriate file permissions so that they are readable by the user invoking Q-Chem.

```
$rem
rem_variable            option  comment
rem_variable            option  comment
...
$end
```

Figure 9.2 Format of the *.qchemrc* and *preferences* files

```
$rem
INCORE_INTS_BUFFER   4000000   More integrals in memory
DIIS_SUBSPACE_SIZE   5         Modify max DIIS subspace size
THRESH                         10
$end
```

Example 9.1 An example of a *.qchemrc* file to apply program default override *$rem* settings to all of the user's Q-Chem jobs.

9.1.2 RECOMMENDATIONS

As mentioned, the customization files are specifically suited for placing long-term machine specific defaults, as clearly some of the defaults placed by Q-Chem will not be optimal on large or very small machines. The following *$rem* variables are examples of those which should be considered, but the user is free to include as few or as many as desired (*INCORE_INTS_BUFFER*, *MEMORY*, *SCF_CONVERGENCE*, *THRESH*, *NBO*).

Q-Chem will print a warning message to advise the user if a *$rem* keyword section has been detected in either *.qchemrc* or *preferences*.

9.2 Q-Chem Auxiliary files ($QCAUX)

The *$QCAUX* environment variable determines the directory where Q-Chem searches for data files and the machine license. This directory defaults to *$QC/aux*. Presently, the *$QCAUX* contains four subdirectories: *atoms*, *basis*, *drivers* and *license*. The *atoms* directory contains data used for the SAD (Chapter 4) SCF density guess; *basis* contains the exponents and contraction coefficients for the standard basis sets available in Q-Chem (Chapter 6); *drivers* contains important information for Q-Chem's AOINTS package and the *license* directory contains the user license. By setting the *$QCAUX* variable, the *aux* directory may be moved to a separate location from the rest of the program, *e.g.*, to save disk space. Alternatively, one may place a soft link in *$QC* to the actual *aux* directory.

Users should not alter any files or directories within *$QCAUX* unless directed by Q-Chem, Inc.

9.3 Additional Q-Chem Output

The following features are under development and users are advised that those presented, and the format requirements to invoke them, are subject to change in future releases.

9.3.1 Plot Intermediates

Visualization is a powerful method of interpreting quantum chemical information. While Q-Chem has a number of supporting GUIs, it is also possible to produce the raw data which can subsequently be passed on to other third party applications. Users are able to direct Q-Chem to output plotting data for molecular orbitals, electron densities and attach/detach analysis. This is achieved by setting *$rem WAVEFUNCTION_ANALYSIS* to 200 and inserting a *$plots* keyword section into the input file using the format depicted in Figure 9.3.

```
$plots
place a comment on the first line
Nx         xmin         xmax
Ny         ymin         ymax
Nz         zmin         zmax
Nmo        Nrho         0       Nda
MO_i       MO_j         MO_k ... (if Nmo > 0)
rho_i      rho_j        rho_k ... (if Nrho > 0)
da_i       da_j         da_k ... (if Nda > 0)
$end
```

Figure 9.3 Format requirements for obtaining raw data for plots.

Line 1 of the *$plots* keyword section is reserved for comments. Lines 2-4 list the number of one dimension points and the range of the grid (note that coordinate ranges are in Angstroms, while all output is in atomic units). Line 5 must contain 4 non-negative integers indicating the number of: molecular orbitals (*Nmo*), electron densities (*Nrho*), (a place-holding zero) and attach/detach densities (*Nda*), to have mesh values calculated.

The final three lines specify which MOs, electron densities and CIS attach/detach states are to be plotted (the line can be left blank, or removed, if the number of items to plot is zero). Molecular orbitals are numbered 1 ... $N\alpha$, $N\alpha$ + 1 ... $N\alpha + N\beta$; electron densities numbered where 1= ground state, 2 = first excited state, 3 = second excited state, *etc*.; and attach/detach specified from state 1→*Nda*.

```
$plots
dump data to create plots of MOs and atc/det analysis
10      -2      2
10      -2      2
10      -2      2
3  0  0  3
5 6 7
1 2 3
$end
```

Example 9.2 Example of a *$plots* section in an input file. This example will provide data to plot MOs 5, 6 & 7 as well as attachment/detachment analysis of CIS states 1, 2 & 3.

All output data is printed to files in the working directory, overwriting any existing file of the same name. Molecular orbital data is printed to a file called "*plot.mo*"; densities to "*plots.hf*"; restricted unrelaxed attachment/detachment analysis to "*plot.attach.alpha*" and "*plot.detach.alpha*"; unrestricted unrelaxed attachment/detachment analysis to "*plot.attach.alpha*", "*plot.detach.alpha*", "*plot.attach.beta*" and "*plot.detach.beta*"; restricted relaxed attachment/detachment analysis to "*plot.attach.rlx.alpha*" and "*plot.detach.rlx.alpha*"; unrestricted relaxed attachment/detachment analysis to "*plot.attach.rlx.alpha*", "*plot.detach.rlx.alpha*", "*plot.attach.rlx.beta*" and "*plot.detach.rlx.beta*". Output is printed in atomic units - coordinates first followed by item value.

```
x1    y1   z1     a1    a2 .... aN
x2    y1   z1     b1    b2 .... bN
...
```

Figure 9.4 File output format for all raw plotting data.

Refer to Chapter 5 regarding the attachment/detachment analysis and using the relaxed CIS density (*CIS_RELAXED_DENSITY*) in attachment/detachment analysis.

9.3.2 Third Party FCHK File

Q-Chem can be instructed to output a third party "fchk" file, "*Test.FChk*", to the working directory by setting the *$rem* variable *GUI* to 2. Please note that for future releases of Q-Chem this feature, and the method used to invoke it, is subject to change.

APPENDIX A GEOMETRY OPTIMIZATION WITH Q-CHEM

A.1 INTRODUCTION

Geometry optimization refers to the determination of stationary points, principally minima and transition states, on molecular potential energy surfaces. It is an iterative process, requiring the repeated calculation of energies, gradients and (possibly) Hessians at each optimization cycle until convergence is attained. The optimization step involves modifying the current geometry, utilizing current and previous energy, gradient and Hessian information to produce a revised geometry which is closer to the target stationary point than its predecessor was. The art of geometry optimization lies in calculating the step **h**, the displacement from the starting geometry on that cycle, so as to converge in as few cycles as possible.

There are four main factors that influence the rate of convergence. These are:

1. initial starting geometry
2. algorithm used to determine the step **h**
3. quality of the Hessian (second derivative) matrix
4. coordinate system chosen

The first of these factors is obvious: the closer the initial geometry is to the final converged geometry the fewer optimization cycles it will take to reach it. The second factor is again obvious: if a poor step **h** is predicted, this will obviously slow down the rate of convergence. The third factor is related to the second: the best algorithms make use of second derivative (curvature) information in calculating **h**, and the better this information is, the better will be the predicted step. The importance of the fourth factor (the coordinate system) has only been generally appreciated relatively recently: a good choice of coordinates can enhance the convergence rate by an order of magnitude (a factor of 10) or more, depending on the molecule being optimized.

Q-Chem includes a powerful suite of algorithms for geometry optimization written by Jon Baker and known collectively as OPTIMIZE. These algorithms have been developed and perfected over the past ten years and the code is robust and has been well tested. OPTIMIZE is a general geometry optimization package for locating both minima and transition states. It can optimize using Cartesian, *Z*-matrix coordinates or delocalized internal coordinates. The last of these are generated automatically from the Cartesian coordinates and are often found to be particularly effective. It also handles fixed constraints on distances, angles, torsions and out-of-plane bends, between any atoms in the molecule, whether or not the desired constraint is satisfied in the starting geometry. Finally it can freeze atomic positions, or any *X*, *Y*, *Z* Cartesian atomic coordinates.

OPTIMIZE is designed to operate with minimal user input. All that is required is the initial guess geometry, either in Cartesian coordinates (*e.g.* from a suitable model builder such as HyperChem) or as a *Z*-matrix, the type of stationary point being sought (minimum or transition state) and details of any imposed constraints. All decisions as to the optimization strategy (what algorithm to use, what coordinate system to choose, how to handle the constraints) are made by OPTIMIZE.

Note particularly, that although the starting geometry is input in a particular coordinate system (as a Z-matrix, for example) these coordinates are not necessarily used during the actual optimization. The best coordinates for the majority of geometry optimizations are delocalized internals, and these will be tried first. Only if delocalized internals fail for some reason, or if conditions prevent them being used (*e.g.*, frozen atoms) will other coordinate systems be tried. If all else fails the default is to switch to Cartesian coordinates. Similar defaults hold for the optimization algorithm, maximum step size, convergence criteria, *etc.* You may of course override the default choices and force a particular optimization strategy, but it is not normally necessary to provide OPTIMIZE with anything other than the minimal information outlined above.

The heart of the OPTIMIZE package (for both minima and transition states) is Baker's EF (Eigenvector Following) algorithm [1]. This was developed following the work of Cerjan and Miller [2] and, Simons and coworkers [3, 4]. The Hessian mode-following option incorporated into this algorithm is capable of locating transition states by walking uphill from the associated minima. By following the lowest Hessian mode, the EF algorithm can locate transition states starting from any reasonable input geometry and Hessian.

An additional option available for minimization is Pulay's GDIIS algorithm [5], which is based on the well known DIIS technique for accelerating SCF convergence [6]. GDIIS must be specifically requested, as the EF algorithm is the default.

Although optimizations can be carried out in Cartesian or *Z*-matrix coordinates, the best choice, as noted above, is usually delocalized internal coordinates. These coordinates were developed very recently by Baker *et al* [7], and can be considered as a further extension of the natural internal coordinates developed by Pulay *et al* [8, 9] and the redundant optimization method of Pulay and Fogarasi [10].

OPTIMIZE incorporates a very accurate and efficient Lagrange multiplier algorithm for constrained optimization. This was originally developed for use with Cartesian coordinates [11, 12] and can handle constraints that are not satisfied in the starting geometry. Very recently the Lagrange multiplier approach has been modified for use with delocalized internals [13]; this is much more efficient and is now the default. The Lagrange multiplier code can locate constrained transition states as well as minima.

A.2 THEORETICAL BACKGROUND

A.2.1 THE NEWTON-RAPHSON STEP

Consider the energy, E($\mathbf{x}_0$) at some point $\mathbf{x}_0$ on a potential energy surface. We can express the energy at a nearby point $\mathbf{x} = \mathbf{x}_0 + \mathbf{h}$ by means of the Taylor series

$$\mathrm{E}(\mathbf{x_0}+\mathbf{h}) = \mathrm{E}(\mathbf{x_0}) + \mathbf{h}^t \frac{\mathrm{d\,E}(\mathbf{x_0})}{\mathrm{d}\,\mathbf{x}} + \tfrac{1}{2}\mathbf{h}^t \frac{\mathrm{d}^2\,\mathrm{E}(\mathbf{x_0})}{\mathrm{d}\,\mathbf{x_1}\,\mathrm{d}\,\mathbf{x_2}}\mathbf{h} + \ldots \tag{A.1}$$

If we knew the exact form of the energy functional E(**x**) and all its derivatives, we could move from the current point $\mathbf{x}_0$ directly to a stationary point, (*i.e.*, we would know exactly what the step **h** ought to be). Since we typically know only the lower derivatives of E(**x**) at best, then we can estimate the step **h** by differentiating the Taylor series with respect to **h**, keeping only the first few terms on the right hand side, and setting the left hand side, dE($\mathbf{x}_0$+**h**)/d**h**, to zero, which is the value it would have at a genuine stationary point. Thus

$$\frac{\mathrm{dE}(\mathbf{x_0}+\mathbf{h})}{\mathrm{d}\,\mathbf{h}} = \frac{\mathrm{dE}(\mathbf{x_0})}{\mathrm{d}\,\mathbf{x}} + \frac{\mathrm{d}^2\mathrm{E}(\mathbf{x_0})}{\mathrm{d}\,\mathbf{x_1}\,\mathrm{d}\,\mathbf{x_2}}\mathbf{h} + \text{higher terms (ignored)} \tag{A.2}$$

From which

$$\mathbf{h} = \mathbf{H}^{-1}\mathbf{g} \tag{A.3}$$

where

$$\frac{\mathrm{dE}}{\mathrm{d}\,\mathbf{x}} \equiv \mathbf{g}\ \text{(gradient vector)},\quad \frac{\mathrm{d}^2\mathrm{E}}{\mathrm{d}\,\mathbf{x_1}\,\mathrm{d}\,\mathbf{x_2}} \equiv \mathbf{H}\ \text{(Hessian matrix)}$$

(A.3) is known as the Newton-Raphson step. It is the major component of almost all geometry optimization algorithms in quantum chemistry.

The above derivation assumed exact first (gradient) and second (Hessian) derivative information. Analytical gradients are available for all methodologies supported in Q-Chem; however analytical second derivatives are not. Furthermore, even if they were, it would not necessarily be advantageous to use them as their evaluation is usually computationally demanding, and, efficient optimizations can in fact be performed without an exact Hessian. An excellent compromise in practice is to begin with an approximate Hessian matrix, and update this using gradient and displacement information generated as the optimization progresses. In this way the starting Hessian can be "improved" at essentially no cost. Using (A.3) with an approximate Hessian is called the quasi Newton-Raphson step.

The nature of the Hessian matrix (in particular its eigenvalue structure) plays a crucial role in a successful optimization. All stationary points on a potential energy surface have a zero gradient vector; however the character of the stationary point (*i.e.*, what type of structure it corresponds to) is determined by the Hessian. Diagonalization of the Hessian matrix can be considered to define a set of mutually orthogonal directions on the energy surface (the eigenvectors) together with the curvature along those directions (the eigenvalues). At a local minimum (corresponding to a well in the potential energy surface) the curvature along all of these directions must be positive, reflecting the fact that a small displacement along any of these directions causes the energy to rise. At a transition state, the curvature is negative (*i.e.*, the energy is a maximum) along one direction, but positive along all the others. Thus, for a stationary point to be a transition state the Hessian matrix at that point must have one and only one negative eigenvalue, while for a minimum the Hessian must have all positive eigenvalues. In the latter case the Hessian is called *positive definite*. If searching for a minimum it is important that the Hessian matrix be positive definite; in fact, unless the Hessian is positive definite there is no guarantee that the step predicted by (A.3) is even a descent step (*i.e.*, a direction that will actually lower the energy). Similarly, for a transition state search, the Hessian must have one negative eigenvalue. Maintaining the Hessian eigenvalue structure is not difficult for minimization, but it can be a difficulty when trying to find a transition state.

In a diagonal Hessian representation the Newton-Raphson step can be written

$$\mathbf{h} = \sum \frac{-F_i}{b_i} \mathbf{u}_i \tag{A.4}$$

where $\mathbf{u}_i$ and b_i are the eigenvectors and eigenvalues of the Hessian matrix $\mathbf{H}$ and $F_i = \mathbf{u}_i^\mathrm{t}\mathbf{g}$ is the component of $\mathbf{g}$ along the local direction (eigenmode) $\mathbf{u}_i$. As discussed by Simons *et al.* [3], the Newton-Raphson step can be considered as minimizing along directions $\mathbf{u}_i$ which have positive eigenvalues and maximizing along directions with negative eigenvalues. Thus, if the user is searching for a minimum and the Hessian matrix is positive definite, then the Newton-Raphson step is appropriate since it is attempting to minimize along all directions simultaneously. However, if the Hessian has one or more negative eigenvalues, then the basic Newton-Raphson step is not appropriate for a minimum search, since it will be maximizing and not minimizing along one or more directions. Exactly the same arguments apply during a transition state search except that the Hessian must have one negative eigenvalue, because the user has to maximize along one direction. However, there must be *only* one negative eigenvalue. A positive definite Hessian is a disaster for a transition state search because the Newton-Raphson step will then lead towards a minimum.

If firmly in a region of the potential energy surface with the right Hessian character, then a careful search (based on the Newton-Raphson step) will almost always lead to a stationary point of the correct type. However, this is only true if the Hessian is exact. If an approximate Hessian is being improved by updating, then there is no guarantee that the Hessian eigenvalue structure will be retained from one cycle to the next unless one is very careful during the update. Updating procedures that "guarantee" conservation of a positive definite Hessian do exist (or at least warn the user if the update is likely to introduce negative eigenvalues). This can be very useful during a minimum search; but there are no such guarantees for preserving the Hessian character (one and only one negative eigenvalue) required for a transition state.

In addition to the difficulties in retaining the correct Hessian character, there is the matter of obtaining a "correct" Hessian in the first instance. This is particularly acute for a transition state search. For a minimum search it is possible to "guess" a reasonable, positive-definite starting Hessian (for example, by carrying out a molecular mechanics minimization initially and using the mechanics Hessian to begin the *ab initio* optimization) but this option is usually not available for transition states. Even if the user calculates the Hessian exactly at the starting geometry, the guess for the structure may not be sufficiently accurate, and the expensive, exact Hessian may not have the desired eigenvalue structure.

Consequently, particularly for a transition state search, an alternative to the basic Newton-Raphson step is clearly needed, especially when the Hessian matrix is inappropriate for the stationary point being sought.

One of the first algorithms that was capable of taking corrective action during a transition state search if the Hessian had the wrong eigenvalue structure, was developed by Poppinger [14], who suggested that, instead of taking the Newton- Raphson step, if the Hessian had all positive eigenvalues, the lowest Hessian mode be followed uphill; whereas, if there were two or more negative eigenvalues, the mode corresponding to the least negative eigenvalue be followed downhill. While this step should lead the user back into the right region of the energy surface, it has the disadvantage that the user is maximizing or minimizing along one mode only, unlike the Newton-Raphson step which maximizes/minimizes along all modes simultaneously. Another drawback is that successive such steps tend to become linearly dependent, which degrades most of the commonly used Hessian updates.

A.2.2 THE EIGENVECTOR FOLLOWING (EF) ALGORITHM

The work of Cerjan and Miller [2], and later Simons and coworkers [3, 4], showed that there was a better step than simply directly following one of the Hessian eigenvectors. A simple modification to the Newton-Raphson step is capable of guiding the search away from the current region towards a stationary point with the required characteristics. This is

$$\mathbf{h} = \sum \frac{-F_i}{(b_i - \lambda)} \mathbf{u}_i \tag{A.5}$$

in which λ can be regarded as a shift parameter on the Hessian eigenvalue b_i. Scaling the Newton-Raphson step in this manner effectively directs the step to lie primarily, but not exclusively (unlike Poppinger's algorithm [14]), along one of the local eigenmodes, depending on the value chosen for λ. References [2-4] all utilize the same basic approach (A.5) but differ in the means of determining the value of λ.

The EF algorithm [1] utilizes the rational function approach presented in [4], yielding an eigenvalue equation of the form

$$\begin{pmatrix} \mathbf{H} & \mathbf{g} \\ \mathbf{g}^t & 0 \end{pmatrix} \begin{pmatrix} \mathbf{h} \\ 1 \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{h} \\ 1 \end{pmatrix} \qquad \text{(A.6)}$$

from which a suitable λ can be obtained. Expanding (A.6) gives

$$(\mathbf{H} - \lambda)\mathbf{h} + \mathbf{g} = 0 \qquad \text{(A.7a)}$$

$$\mathbf{g}^t\mathbf{h} = \lambda \qquad \text{(A.7b)}$$

In terms of a diagonal Hessian representation, (A.7a) rearranges to (A.5), and substitution of (A.5) into the diagonal form of (A.7b) gives

$$\sum \frac{-F_i^2}{(b_i - \lambda)} = \lambda \qquad \text{(A.8)}$$

which can be used to evaluate λ iteratively.

The eigenvalues, λ, of the RFO equation (A.6) have the following important properties [4]:

1. The ($n+1$) values of λ bracket the n eigenvalues of the Hessian matrix $\lambda_i < b_i < \lambda_{i+1}$
2. At a stationary point, one of the eigenvalues, λ, of (A.6) is zero and the other *n* eigenvalues are those of the Hessian at the stationary point.
3. For a saddle point of order *m*, the zero eigenvalue separates the *m* negative and the (*n-m*) positive Hessian eigenvalues.

This last property, the separability of the positive and negative Hessian eigenvalues, enables two shift parameters to be used, one for modes along which the energy is to be maximized and the other for which it is minimized. For a transition state (a first-order saddle point), in terms of the Hessian eigenmodes, we have the two matrix equations

$$\begin{pmatrix} b_1 & F_1 \\ F_1 & 0 \end{pmatrix} \begin{pmatrix} h_1 \\ 1 \end{pmatrix} = \lambda_p \begin{pmatrix} h_1 \\ 1 \end{pmatrix} \qquad \text{(A.9a)}$$

$$\begin{pmatrix} b_2 & & & F_2 \\ & \ddots & \mathbf{0} & \vdots \\ & \mathbf{0} & b_n & F_n \\ F_2 & \cdots & F_n & 0 \end{pmatrix} \begin{pmatrix} h_2 \\ \vdots \\ h_n \\ 1 \end{pmatrix} = \lambda_n \begin{pmatrix} h_2 \\ \vdots \\ h_n \\ 1 \end{pmatrix} \tag{A.9b}$$

where it is assumed that we are maximizing along the lowest Hessian mode $\mathbf{u}_1$. Note that λ_p is the highest eigenvalue of (A.9a) (it is always positive and approaches zero at convergence) and λ_n is the lowest eigenvalue of (A.9b) (it is always negative and again approaches zero at convergence).

Choosing these values of λ gives a step that attempts to maximize along the lowest Hessian mode, while at the same time minimizing along all the other modes. It does this regardless of the Hessian eigenvalue structure (unlike the Newton-Raphson step). The two shift parameters are then used in (A.5) to give the final step

$$\mathbf{h} = \frac{-F_1}{(b_1 - \lambda_p)} \mathbf{u}_1 - \sum_{i=2}^{n} \frac{-F_i}{(b_i - \lambda_n)} \mathbf{u}_i \tag{A.10}$$

If this step is greater than the maximum allowed, it is scaled down. For minimization only one shift parameter, λ_n, would be used which would act on all modes.

In (A.9a) and (A.9b) it was assumed that the step would maximize along the lowest Hessian mode, b_1, and minimize along all the higher modes. However, it is possible to maximize along modes other than the lowest, and in this way perhaps locate transition states for alternative rearrangements/dissociations from the same initial starting point. For maximization along the k^{th} mode (instead of the lowest), (A.9a) is replaced by

$$\begin{pmatrix} b_k & F_k \\ F_k & 0 \end{pmatrix} \begin{pmatrix} h_k \\ 1 \end{pmatrix} = \lambda_p \begin{pmatrix} h_k \\ 1 \end{pmatrix} \tag{A.11}$$

and (A.9b) would now exclude the k^{th} mode but include the lowest. Since what was originally the k^{th} mode is the mode along which the negative eigenvalue is required, then this mode will eventually become the lowest mode at some stage of the optimization. To ensure that the original mode is being followed smoothly from one cycle to the next, the mode that is actually followed is the one with the greatest overlap with the mode followed on the previous cycle. This procedure is known as *mode following*. For more details and some examples, see [1].

A.3 DELOCALIZED INTERNAL COORDINATES

The choice of coordinate system can have a major influence on the rate of convergence during a geometry optimization. For complex potential energy surfaces with many stationary points, a different choice of coordinates can result in convergence to a different final structure.

The key attribute of a good set of coordinates for geometry optimization is the degree of coupling between the individual coordinates. In general, the less coupling the better, as variation of one particular coordinate will then have minimal impact on the other coordinates. Coupling manifests itself primarily as relatively large partial derivative terms between different coordinates. For example, a strong harmonic coupling between two different coordinates, *i* and *j*, results in a large off-diagonal element, H_{ij}, in the Hessian (second derivative) matrix. Normally this is the only type of coupling that can be directly "observed" during an optimization, as third and higher derivatives are ignored in almost all optimization algorithms.

In the early days of computational quantum chemistry geometry optimizations were carried out in Cartesian coordinates. Cartesians are an obvious choice as they can be defined for all systems and gradients and second derivatives are calculated directly in Cartesian coordinates. Unfortunately, Cartesians normally make a poor coordinate set for optimization as they are heavily coupled. Recently, Cartesians have been returning to favour because of their very general nature, and because it has been clearly demonstrated that if reliable second derivative information is available (*i.e.*, a good starting Hessian) and the initial geometry is reasonable, then Cartesians can be as efficient as any other coordinate set for small to medium-sized molecules [15, 16]. Without good Hessian data, however, Cartesians are inefficient, especially for long chain acyclic systems.

In the 1970s Cartesians were replaced by *Z*-matrix coordinates. Initially the *Z*-matrix was utilized simply as a means of geometry input; it is far easier to describe a molecule in terms of bond lengths, bond angles and dihedral angles (the natural way a chemist thinks of molecular structure) than to develop a suitable set of Cartesian coordinates. It was subsequently found that optimization was generally more efficient in *Z*-matrix coordinates than in Cartesians, especially for acyclic systems. This is not always the case, and care must be taken in constructing a suitable *Z*-matrix. A good general rule is ensure that each variable is defined in such a way that changing its value will not change the values of any of the other variables. A brief discussion concerning good *Z*-matrix construction strategy is given by Schlegel [17].

In 1979 Pulay *et al.* published a key paper, introducing what were termed natural internal coordinates into geometry optimization [8]. These coordinates involve the use of individual bond displacements as stretching coordinates, but linear combinations of bond angles and torsions as deformational coordinates. Suitable linear combinations of bends and torsions (the two are considered separately) are selected using group theoretical arguments based on local pseudosymmetry. For example, bond angles around an sp^3 hybridized carbon atom are all approximately tetrahedral, regardless of the groups attached, and idealized tetrahedral symmetry can be used to generate deformational coordinates around the central carbon atom.

The major advantage of natural internal coordinates in geometry optimization is their ability to significantly reduce the coupling, both harmonic and anharmonic, between the various coordinates. Compared to natural internals, *Z*-matrix coordinates arbitrarily omit some angles and torsions (to prevent redundancy), and this can induce strong anharmonic coupling between the coordinates, especially with a poorly constructed *Z*-matrix. Another advantage of the reduced coupling is that successful minimizations can be carried out in natural internals with only an approximate (*e.g.*, diagonal) Hessian provided at the starting geometry. A good starting Hessian is still needed for a transition state search.

Despite their clear advantages, natural internals have only become used widely more recently. This is because, when used in the early programs, it was necessary for the user to define them. This situation changed in 1992 with the development of computational algorithms capable of automatically generating natural internals from input Cartesians [9]. For minimization, natural internals have become the coordinates of first choice [9, 16].

There are some disadvantages to natural internal coordinates as they are commonly constructed and used:

1. Algorithms for the automatic construction of natural internals are complicated. There are a large number of structural possibilities, and to adequately handle even the most common of them can take several thousand lines of code.
2. For the more complex molecular topologies, most assigning algorithms generate more natural internal coordinates than are required to characterize all possible motions of the system (*i.e.*, the generated coordinate set contains redundancies).
3. In cases with a very complex molecular topology (*e.g.*, multiply fused rings and cage compounds) the assigning algorithm may be unable to generate a suitable set of coordinates.

The redundancy problem has recently been addressed in an excellent paper by Pulay and Fogarasi [10], who have developed a scheme for carrying out geometry optimization directly in the redundant coordinate space.

Very recently, Baker *et al.* have developed a set of delocalized internal coordinates [7] which eliminate all of the above-mentioned difficulties. Building on some of the ideas in the redundant optimization scheme of Pulay and Fogarasi [10], delocalized internals form a complete, non-redundant set of coordinates which are as good as, if not superior to, natural internals, and which can be generated in a simple and straightforward manner for essentially any molecular topology, no matter how complex.

Consider a set of *n* internal coordinates $\mathbf{q} = (q_1, q_2, \ldots q_n)^t$. Displacements $\Delta\mathbf{q}$ in $\mathbf{q}$ are related to the corresponding Cartesian displacements $\Delta\mathbf{X}$ by means of the usual B-matrix [18]

$$\Delta\mathbf{q} = \mathbf{B}\Delta\mathbf{X} \tag{A.12}$$

If any of the internal coordinates **q** are redundant, then the rows of the B-matrix will be linearly dependent.

Delocalized internal coordinates are obtained simply by constructing and diagonalizing the matrix $\mathbf{G} = \mathbf{B}\mathbf{B}^t$. Diagonalization of **G** results in two sets of eigenvectors; a set of *m* (typically 3*N*-6, where *N* is the number of atoms) eigenvectors with eigenvalues $\lambda > 0$, and a set of *n-m* eigenvectors with eigenvalues $\lambda = 0$ (to numerical precision). In this way, any redundancies present in the original coordinate set **q** are isolated (they correspond to those eigenvectors with zero eigenvalues). The eigenvalue equation of **G** can thus be written

$$\mathbf{G}(\mathbf{UR}) = (\mathbf{UR})\begin{pmatrix} \Lambda & 0 \\ 0 & 0 \end{pmatrix} \tag{A.13}$$

where **U** is the set of non-redundant eigenvectors of **G** (those with $\lambda > 0$) and **R** is the corresponding redundant set.

The nature of the original set of coordinates **q** is unimportant, as long as it spans all the degrees of freedom of the system under consideration. We include in **q**, all bond stretches, all planar bends and all proper torsions that can be generated based on the atomic connectivity. These individual internal coordinates are termed *primitives*. This blanket approach generates far more primitives than are necessary, and the set **q** contains much redundancy. This is of little concern, as solution of (A.13) takes care of all redundancies.

Note that eigenvectors in both **U** and **R** will each be linear combinations of potentially all the original primitives. Despite this apparent complexity, we take the set of non-redundant vectors **U** as our working coordinate set. Internal coordinates so defined are much more delocalized than natural internal coordinates (which are combinations of a relatively small number of bends or torsions) hence, the term delocalized internal coordinates.

It may appear that because delocalized internals are such a complicated mixing of the original primitive internals, they are a poor choice for use in an actual optimization. On the contrary, arguments can be made that delocalized internals are, in fact, the "best" possible choice, certainly at the starting geometry. The interested reader is referred to the original literature for more details [7].

The situation for geometry optimization, comparing Cartesian, *Z*-matrix and delocalized internal coordinates, and assuming a "reasonable" starting geometry, is as follows:

1. For small or very rigid medium-sized systems (up to about 15 atoms), optimizations in Cartesian and internal coordinates ("good" *Z*-matrix or delocalized internals) should perform similarly.

2. For medium-sized systems (say 15-30 atoms) optimizations in Cartesians should perform as well as optimizations in internal coordinates, provided a reliable starting Hessian is available.
3. For large systems (30+ atoms), unless these are very rigid, neither Cartesian nor *Z*-matrix coordinates can compete with delocalized internals, even with good quality Hessian information. As the system increases, and with less reliable starting geometries, the advantage of delocalized internals can only increase.

There is one particular situation in which Cartesian coordinates may be the best choice. Natural internal coordinates (and by extension delocalized internals) show a tendency to converge to low energy structures [16]. This is because steps taken in internal coordinate space tend to be much larger when translated into Cartesian space, and, as a result, higher energy local minima tend to be “jumped over”, especially if there is no reliable Hessian information available (which is generally not needed for a successful optimization). Consequently, if the user is looking for a local minimum (*i.e.*, a metastable structure) and has both a good starting geometry and a decent Hessian, the user should carry out the optimization in Cartesian coordinates.

A.4 CONSTRAINED OPTIMIZATION

A.4.1 CARTESIAN COORDINATES

Constrained optimization refers to the optimization of molecular structures in which certain parameters (*e.g*., bond lengths, bond angles or dihedral angles) are fixed. In quantum chemistry calculations, this has traditionally been accomplished using *Z*-matrix coordinates, with the desired parameter set in the *Z*-matrix and simply omitted from the optimization space. In 1992, Baker presented an algorithm for constrained optimization directly in Cartesian coordinates [11]. Baker's algorithm used both penalty functions and the classical method of Lagrange multipliers [19], and was developed in order to impose constraints on a molecule obtained from a graphical model builder as a set of Cartesian coordinates. Some improvements widening the range of constraints that could be handled were made in 1993 [12]. Q-Chem includes the latest version of this algorithm, which has been modified to handle constraints directly in delocalized internal coordinates [13].

The essential problem in constrained optimization is to minimize a function of, for example, *n* variables $F(\mathbf{x})$ subject to a series of m constraints of the form $C_i(\mathbf{x}) = 0$, i=l ... m. Assuming $m < n$, then perhaps the best way to proceed (if this were possible in practice) would be to use the m constraint equations to eliminate m of the variables, and then solve the resulting unconstrained problem in terms of the (n-m) independent variables. This is exactly what occurs in a *Z*-matrix optimization. Such an approach cannot be used in Cartesian coordinates as standard distance and angle constraints are non-linear functions of the appropriate coordinates. For example a distance constraint (between atoms *i* and *j* in a molecule) is given in Cartesians by $(R_{ij} - R_0) = 0$, with

$$R_{ij} = \sqrt{\left(x_i - x_j\right)^2 + \left(y_i - y_j\right)^2 + \left(z_i - z_j\right)^2} \tag{A.14}$$

and R_0 the constrained distance. This obviously cannot be satisfied by elimination. What can be eliminated in Cartesians are the individual *x*, *y* and *z* coordinates themselves and in this way individual atoms can be totally or partially frozen.

Internal constraints can be handled in Cartesian coordinates by introducing the Lagrangian function

$$L(\mathbf{x},\lambda) = F(\mathbf{x}) - \sum_{i=1}^{m} \lambda_i C_i(\mathbf{x}) \tag{A.15}$$

which replaces the function $F(\mathbf{x})$ in the unconstrained case. Here, the λ_i are the so-called Lagrange multipliers, one for each constraint $C_i(\mathbf{x})$. Differentiating (A.15) with respect to $\mathbf{x}$ and λ gives

$$\frac{\mathrm{d}\,L(\mathbf{x},\lambda)}{\mathrm{d}\,x_j} = \frac{\mathrm{d}\,F(\mathbf{x})}{\mathrm{d}\,x_j} - \sum_{i=1}^{m} \lambda_i \frac{\mathrm{d}\,C_i(\mathbf{x})}{\mathrm{d}\,x_j} \tag{A.16a}$$

and

$$\frac{\mathrm{d}\,L(\mathbf{x},\lambda)}{\mathrm{d}\lambda_i} = -C_i(\mathbf{x}) \tag{A.16b}$$

At a stationary point of the Lagrangian we have ∇**L** = 0, *i.e.*, all dL/dx_j = 0 and all dL/dλ_i = 0. This latter condition means that all $C_i(\mathbf{x}) = 0$ and thus all constraints are satisfied. Hence, finding a set of values (**x**, λ) for which ∇**L** = 0 will give a possible solution to the constrained optimization problem in exactly the same way as finding an **x** for which **g** = ∇**F** = 0 gives a solution to the corresponding unconstrained problem.

The Lagrangian second derivative matrix, the equivalent of the Hessian matrix in an unconstrained optimization, is given by

$$\nabla^2\mathbf{L} = \begin{pmatrix} \mathrm{d}^2\,L(\mathbf{x},\lambda)/\mathrm{d}x_j\,\mathrm{d}x_k & \mathrm{d}^2\,L(\mathbf{x},\lambda)/\mathrm{d}x_j\,\mathrm{d}\lambda_i \\ \mathrm{d}^2\,L(\mathbf{x},\lambda)/\mathrm{d}x_j\,\mathrm{d}\lambda_i & \mathrm{d}^2\,L(\mathbf{x},\lambda)/\mathrm{d}\lambda_j\,\mathrm{d}\lambda_i \end{pmatrix} \tag{A.17}$$

where

$$\mathrm{d}^2\,L(\mathbf{x},\lambda)/\mathrm{d}x_j\,\mathrm{d}x_k = \mathrm{d}^2\,F(\mathbf{x})/\mathrm{d}x_j\,\mathrm{d}x_k - \sum \lambda_i\,\mathrm{d}^2\,F(\mathbf{x})/\mathrm{d}x_j\,\mathrm{d}x_k \tag{A.17a}$$

$$\mathrm{d}^2\,L(\mathbf{x},\lambda)/\mathrm{d}x_j\,\mathrm{d}\lambda_i = -\mathrm{d}\,C_i(\mathbf{x})/\mathrm{d}x_j \tag{A.17b}$$

$$\mathrm{d}^2\,L(\mathbf{x},\lambda)/\mathrm{d}\lambda_j\,\mathrm{d}\lambda_i = 0 \tag{A.17c}$$

Thus in addition to the standard gradient vector and Hessian matrix for the unconstrained function *F*(**x**), we need both the first and second derivatives (with respect to coordinate displacement) of the constraint functions. Once these quantities are available, the corresponding Lagrangian gradient, given by (A.16), and Lagrangian second derivative matrix, given by (A.17), can be formed, and the optimization step calculated in a similar manner to that for a standard unconstrained optimization [11].

In the Lagrange multiplier method, the unknown multipliers, λ_i, are an integral part of the parameter set. This means that the optimization space consists of all *n* variables **x** plus all *m* Lagrange multipliers λ, one for each constraint. The total dimension of the constrained optimization problem, *n+m*, has thus increased by *m* compared to the corresponding unconstrained case. The Lagrangian Hessian matrix, ∇^2**L**, has *m* extra modes compared to the standard (unconstrained) Hessian matrix, ∇^2**F**. What normally happens is that these additional modes are dominated by the constraints (*i.e.*, their largest components correspond to the constraint Lagrange multipliers) and they have negative curvature (a negative Hessian eigenvalue). This is perhaps not surprising when one realizes that any motion in the parameter space that breaks the constraints is likely to lower the energy.

Compared to a standard unconstrained minimization, where a stationary point is sought at which the Hessian matrix has all positive eigenvalues, in the constrained problem we are looking for a stationary point of the Lagrangian function at which the Lagrangian Hessian matrix has as many negative eigenvalues as there are constraints (*i.e.*, we are looking for an m^{th} order saddle point). For further details and practical applications of constrained optimization using Lagrange multipliers in Cartesian coordinates, see [11].

Eigenvector following can be implemented in a constrained optimization in a similar way to the unconstrained case. Considering a constrained minimization with *m* constraints, then (A.9a) is replaced by

$$\begin{pmatrix} b_1 & & & F_1 \\ & \ddots & \mathbf{0} & \vdots \\ & \mathbf{0} & b_m & F_m \\ F_1 & \cdots & F_m & 0 \end{pmatrix} \begin{pmatrix} h_1 \\ \vdots \\ h_m \\ 1 \end{pmatrix} = \lambda_p \begin{pmatrix} h_1 \\ \vdots \\ h_m \\ 1 \end{pmatrix} \qquad \text{(A.18a)}$$

and (A.9b) by

$$\begin{pmatrix} b_{m+1} & & & F_{m+1} \\ & \ddots & \mathbf{0} & \vdots \\ & \mathbf{0} & b_{m+n} & F_{m+n} \\ F_{m+1} & \cdots & F_{m+n} & 0 \end{pmatrix} \begin{pmatrix} h_{m+1} \\ \vdots \\ h_{m+n} \\ 1 \end{pmatrix} = \lambda_n \begin{pmatrix} h_{m+1} \\ \vdots \\ h_{m+n} \\ 1 \end{pmatrix} \qquad \text{(A.18b)}$$

where now the b_i are the eigenvalues of $\nabla^2\mathbf{L}$, with corresponding eigenvectors $\mathbf{u}_i$, and $F_i = \mathbf{u}_i^t\nabla\mathbf{L}$. Here (A.18a) includes the *m* constraint modes along which a negative Lagrangian Hessian eigenvalue is required, and (A.18b) includes all the other modes.

Equations (A.18a) and (A.18b) implement eigenvector following for a constrained minimization. Constrained transition state searches can be carried out by selecting one extra mode to be maximized in addition to the *m* constraint modes, *i.e.*, by searching for a saddle point of the Lagrangian function of order *m*+l.

It should be realized that, in the Lagrange multiplier method, the desired constraints are only satisfied at convergence, and not necessarily at intermediate geometries. The Lagrange multipliers are part of the optimization space; they vary just as any other geometrical parameter and, consequently the degree to which the constraints are satisfied changes from cycle to cycle, approaching 100% satisfied near convergence. One advantage this brings is that, unlike in a standard *Z*-matrix approach, constraints do not have to be satisfied in the starting geometry.

Imposed constraints can normally be satisfied to very high accuracy, 10^{-6} or better. However, problems can arise for both bond and dihedral angle constraints near 0° and 180° and, instead of attempting to impose a single constraint, it is better to split angle constraints near these limiting values into two by using a dummy atom [12], exactly analogous to splitting a 180° bond angle into two 90° angles in a *Z*-matrix.

Note: Exact 0° and 180° single angle constraints cannot be imposed, as the corresponding constraint normals, $\nabla \mathbf{C}_i$, are zero, and would result in rows and columns of zeros in the Lagrangian Hessian matrix.

A.4.2 DELOCALIZED INTERNAL COORDINATES

We do not give further details of the optimization algorithms available in Q-Chem for imposing constraints in Cartesian coordinates, as it is far simpler and easier to do this directly in delocalized internal coordinates.

At first sight it does not seem particularly straightforward to impose any constraints at all in delocalized internals, given that each coordinate is potentially a linear combination of all possible primitives. However, this is deceptive, and in fact all standard constraints can be imposed by a relatively simple Schmidt orthogonalization procedure. In this instance consider a unit vector with unit component corresponding to the primitive internal (stretch, bend or torsion) that one wishes to keep constant. This vector is then projected on to the full set, **U**, of active delocalized coordinates, normalized, and then all *n*, for example, delocalized internals are Schmidt orthogonalized in turn to this normalized, projected constraint vector. The last coordinate taken in the active space should drop out (since it will be linearly dependent on the other vectors and the constraint vector) leaving (*n*-1) active vectors and one constraint vector.

In more detail, the procedure is as follows (taken directly from [7]). The initial (usually unit) constraint vector **C** is projected on to the set **U** of delocalized internal coordinates according to

$$\mathbf{C}^{proj} = \sum \langle \mathbf{C} | \mathbf{U}_k \rangle \mathbf{U}_k \qquad \text{(A.19)}$$

where the summation is over all n active coordinates $\mathbf{U}_k$. The projected vector $\mathbf{C}^{proj}$ is then normalized and an (*n*+l) dimensional vector space **V** is formed, comprising the normalized, projected constraint vector together with all active delocalized coordinates

$$\mathbf{V} = \left\{ \mathbf{C}^{proj}, \mathbf{U}_k \;\; k = 1 \ldots n \right\} \qquad \text{(A.20)}$$

This set of vectors is Schmidt orthogonalized according to the standard procedure

$$\tilde{\mathbf{V}}_k = \alpha_k \left(\mathbf{V}_k - \sum_{l=1}^{k-1} \langle \mathbf{V}_k | \tilde{\mathbf{V}}_l \rangle \tilde{\mathbf{V}}_l \right) \qquad \text{(A.21)}$$

where the first vector taken, $\mathbf{V}_1$, is $\mathbf{C}^{proj}$. The α_k in (A.21) is a normalization factor. As noted above, the last vector taken, $\mathbf{V}_{n+1} = \mathbf{U}_k$, will drop out, leaving a fully orthonormal set of (*n*-1) active vectors and one constraint vector.

After the Schmidt orthogonalization the constraint vector will contain all the weight in the active space of the primitive to be fixed, which will have a zero component in all of the other (*n*-1) vectors. The fixed primitive has thus been isolated entirely in the constraint vector which can now be removed from the active subspace for the geometry optimization step.

Extension of the above procedure to multiple constraints is straightforward. In addition to constraints on individual primitives, it is also possible to impose combinatorial constraints. For example, if, instead of a unit vector, one started the constraint procedure with a vector in which two components were set to unity, then this would impose a constraint in which the sum of the two relevant primitives were always constant. In theory any desired linear combination of any primitives could be constrained.

Note further that imposed constraints are not confined to those primitive internals generated from the initial atomic connectivity. If we wish to constrain a distance, angle or torsion between atoms that are not formally connected, then all we need to do is add that particular coordinate to our primitive set. It can then be isolated and constrained in exactly the same way as a formal connectivity constraint.

Everything discussed thus far regarding the imposition of constraints in delocalized internal coordinates has involved isolating each constraint in one vector which is then eliminated from the optimization space. This is very similar in effect to a *Z*-matrix optimization, in which constraints are imposed by elimination. This, of course, can only be done if the desired constraint is satisfied in the starting geometry. We have already seen that the Lagrange multiplier algorithm, used to impose distance, angle and torsion constraints in Cartesian coordinates, can be used even when the constraint is not satisfied initially. The Lagrange multiplier method can also be used with delocalized internals, and its implementation with internal coordinates brings several simplifications and advantages.

In Cartesians, as already noted, standard internal constraints (bond distances, angles and torsions) are somewhat complicated non-linear functions of the *x*, *y* and *z* coordinates of the atoms involved. A torsion, for example, which involves four atoms, is a function of twelve different coordinates. In internals, on the other hand, each constraint is a coordinate in its own right and is therefore a simple linear function of just one coordinate (itself).

If we denote a general internal coordinate by *R*, then the constraint function $C_\mathrm{i}(\mathbf{R})$ is a function of one coordinate, R_i, and it and its derivatives can be written

$$C_i(R_i) = R_i - R_0 \qquad \text{(A.22a)}$$

$$\mathrm{d}\,C_i(R_i)\,/\,\mathrm{d}\,R_i = 1;\;\; \mathrm{d}\,C_i(R_i)\,/\,\mathrm{d}\,R_j = 0 \qquad \text{(A.22b)}$$

$$\mathrm{d}^2\,C_i(R_i)\,/\,\mathrm{d}\,R_i\,\mathrm{d}\,R_j = 0 \qquad \text{(A.22c)}$$

where in (A.22a), R_0 is the desired value of the constrained coordinate, and R_i is its current value. From (A.22b) we see that the constraint normals, $\mathrm{d}C_\mathrm{i}(\mathbf{R})/\mathrm{d}R_i$, are simply unit vectors and the Lagrangian Hessian matrix, (A.17), can be obtained from the normal Hessian matrix by adding *m* columns (and *m* rows) of, again, unit vectors.

A further advantage, in addition to the considerable simplification, is the handling of 0° and 180° dihedral angle constraints. In Cartesian coordinates it is not possible to formally constrain bond angles and torsions to exactly 0° or 180° because the corresponding constraint normal is a zero vector. Similar difficulties do not arise in internal coordinates, at least for torsions, because the constraint normals are unit vectors regardless of the value of the constraint; thus 0° and 180° dihedral angle constraints can be imposed just as easily as any other value. 180° bond angles still cause difficulties, but near-linear arrangements of atoms require special treatment even in unconstrained optimizations; a typical solution involves replacing a near 180° bond angle by two special linear co-planar and perpendicular bends [20], and modifying the torsions where necessary. A linear arrangement can be enforced by constraining the co-planar and perpendicular bends.

One other advantage over Cartesians is that in internals the constraint coordinate can be eliminated once the constraint is satisfied to the desired accuracy (the default tolerance is 10^{-6} in atomic units: bohrs and radians). This is not possible in Cartesians due to the functional form of the constraint. In Cartesians, therefore, the Lagrange multiplier algorithm must be used throughout the entire optimization, whereas in delocalized internal coordinates it need only be used until all desired constraints are satisfied; as constraints become satisfied they can simply be eliminated from the optimization space and once all constraint coordinates have been eliminated standard algorithms can be used in the space of the remaining unconstrained coordinates. Normally, unless the starting geometry is particularly poor in this regard, constraints are satisfied fairly early on in the optimization (and at more or less the same time for multiple constraints), and Lagrange multipliers only need to be used in the first half-dozen or so cycles of a constrained optimization in internal coordinates.

A.5 GDIIS

Direct inversion in the iterative subspace (DIIS) was originally developed by Pulay for accelerating SCF convergence [6]. Subsequently, Csaszar and Pulay used a similar scheme for geometry optimization, which they termed GDIIS [5]. The method is somewhat different from the usual quasi-Newton type approach and is included in OPTIMIZE as an alternative to the EF algorithm. Tests indicate that its performance is similar to EF, at least for small systems; however there is rarely an advantage in using GDIIS in preference to EF.

In GDIIS, geometries ($\mathbf{x}_i$) generated in previous optimization cycles are linearly combined to find the "best" geometry on the current cycle

$$\mathbf{x}_n = \sum_{i=1}^{m} c_i \mathbf{x}_i \tag{A.23}$$

where the problem is to find the best values for the coefficients c_i.

If we express each geometry, $\mathbf{x}_i$, by its deviation from the sought-after final geometry, $\mathbf{x}_f$, *i.e.*, $\mathbf{x}_f = \mathbf{x}_i + \mathbf{e}_i$, where $\mathbf{e}_i$ is an error vector, then it is obvious that if the conditions

$$\mathbf{r} = \sum c_i \mathbf{e}_i \tag{A.24}$$

and

$$\sum c_i = 1 \tag{A.25}$$

are satisfied, then the relation

$$\sum c_i \mathbf{x}_i = \mathbf{x}_f \tag{A.26}$$

also holds.

The true error vectors $\mathbf{e}_i$ are, of course, unknown. However, in the case of a nearly quadratic energy function they can be approximated by

$$\mathbf{e}_i = -\mathbf{H}^{-1}\mathbf{g}_i \tag{A.27}$$

where $\mathbf{g}_i$ is the gradient vector corresponding to the geometry $\mathbf{x}_i$ and $\mathbf{H}$ is an approximation to the Hessian matrix. Minimization of the norm of the residuum vector $\mathbf{r}$, (A.24), together with the constraint equation, (A.25), leads to a system of (m+l) linear equations

$$\begin{pmatrix} B_{11} & \cdots & B_{1m} & 1 \\ \vdots & \ddots & \vdots & \vdots \\ B_{m1} & \cdots & B_{mm} & 1 \\ 1 & \cdots & 1 & 0 \end{pmatrix} \begin{pmatrix} c_1 \\ \vdots \\ c_m \\ -\lambda \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix} \tag{A.28}$$

where $B_{ij} = \langle \mathbf{e}_i | \mathbf{e}_j \rangle$ is the scalar product of the error vectors $\mathbf{e}_i$ and $\mathbf{e}_j$, and λ is a Lagrange multiplier.

The coefficients c_i determined from (A.28) are used to calculate an intermediate interpolated geometry

$$\mathbf{x}'_{m+1} = \sum c_i \mathbf{x}_i \tag{A.29}$$

and its corresponding interpolated gradient

$$\mathbf{g}'_{m+1} = \sum c_i \mathbf{g}_i \tag{A.30}$$

A new, independent geometry is generated from the interpolated geometry and gradient according to

$$\mathbf{x}_{m+1} = \mathbf{x}^{'}_{m+1} - \mathbf{H}^{-1}\mathbf{g}^{'}_{m+1} \tag{A.31}$$

Note: Convergence is theoretically guaranteed regardless of the quality of the Hessian matrix (as long as it is positive definite), and the original GDIIS algorithm used a static Hessian (*i.e*. the original starting Hessian, often a simple unit matrix, remained unchanged during the entire optimization). However, updating the Hessian at each cycle generally results in more rapid convergence, and this is the default in OPTIMIZE.

Other modifications to the original method include limiting the number of previous geometries used in (A.23) and, subsequently, by neglecting earlier geometries, and eliminating any geometries more than a certain distance (default 0.3 au) from the current geometry.

A.6 References

[1] J. Baker, *J. Comp. Chem.* (1986), **7**, 385.

[2] C. J. Cerjan and W. H. Miller, *J. Chem. Phys.* (1981), **75**, 2800.

[3] J. Simons, P. Jorgensen, H. Taylor and J. Ozment, *J. Phys. Chem.* (1983), **87,** 2745.

[4] A. Banerjee, N. Adams, J. Simons and R. Shepard, *J. Phys. Chem.* (1985), **89**, 52.

[5] P. Csaszar and P. Pulay, *J. Mol. Struct. Theochem.* (1984), **114**, 31.

[6] P. Pulay, *J. Comp. Chem.* (1982), **3**, 556.

[7] J. Baker, A. Kessi and B. Delley, *J. Chem. Phys.* (1996), **105**, 192.

[8] P. Pulay, G. Fogarasi, F. Pang and J. E. Boggs, *J. Am. Chem. Soc.* (1979), **101**, 2550.

[9] G. Fogarasi, X. Zhou, P. W. Taylor and P. Pulay, *J. Am. Chem. Soc.* (1992), **114**, 8191.

[10] P. Pulay and G. Fogarasi, *J. Chem. Phys.* (1992), **96**, 2856.

[11] J. Baker, *J. Comp. Chem.* (1992), **13**, 240.

[12] J. Baker and D. Bergeron, *J. Comp. Chem.* (1993), **14**, 1339.

[13] J. Baker, *J. Comp. Chem.* in press.

[14] D. Poppinger, *Chem. Phys. Letts.* (1975), **35**, 550.

[15] J. Baker and W. J. Hehre, *J. Comp. Chem.* (1991), **12**, 606.

[16] J. Baker, *J. Comp. Chem.* (1993), **14**, 1085.

[17] H. B. Schlegel, *Theor. Chim. Acta.* (1984), **66**, 333.

[18] E. B. Wilson, J. C. Decius and P. C. Cross, *Molecular Vibrations,* McGraw-Hill, New York, 1955.

[19] R. Fletcher, *Practical Methods of Optimization: vol. 2 - Constrained Optimization,* Wiley, New York, 1981.

[20] S. Califano *Vibrational States*, Wiley, London, 1976.

APPENDIX B AOINTS

B.1 INTRODUCTION

Within the Q-Chem program, an Atomic Orbital INTegralS (AOINTS) package has been developed which, while relatively invisible to the user, is one of the keys to the overall speed and efficiency of the Q-Chem program.

> "Ever since Boys' introduction of Gaussian basis sets to quantum chemistry in 1950, the calculation and handling of the notorious two-electron-repulsion integrals (ERIs) over Gaussian functions has been an important avenue of research for practicing computational chemists. Indeed, the emergence of practically useful computer programs ... has been fuelled in no small part by the development of sophisticated algorithms to compute the very large number of ERIs that are involved in calculations on molecular systems of even modest size." [1].

The ERI engine of any competitive quantum chemistry software package will be one of the most complicated aspects of the package as whole. Coupled with the importance of such an engine's efficiency, a useful yardstick of a program's anticipated performance can be quickly measured by considering the components of its ERI engine. In recent times, developers at Q-Chem, Inc. have made significant contributions to the advancement of ERI algorithm technology (for example see [1-10]), and it is not surprising that Q-Chem's AOINTS package is considered the most advanced of its kind.

B.2 HISTORICAL PERSPECTIVE

Prior to the 1950's, the most difficult step in the systematic application of Schrödinger wave mechanics to chemistry was the calculation of the notorious two-electron integrals that measure the repulsion between electrons. Boys [11] showed that this step can be made easier (although still time consuming) if Gaussian, rather than Slater, orbitals are used in the basis set. Following the landmark paper of computational chemistry [12] (again due to Boys) programs were constructed that could calculate all the ERIs that arise in the treatment of a general polyatomic molecule with *s* and *p* orbitals. However, the programs were painfully slow and could only be applied to the smallest of molecular systems.

In 1969, Pople constructed a breakthrough ERI algorithm, a hundred time faster than its predecessors. The algorithm remains the fastest available for its associated integral classes and is now referred to as the Pople-Hehre axis-switch method [13].

Over the two decades following Pople's initial development, an enormous amount of research effort into the construction of ERIs was documented, which built on Pople's original success. Essentially, the advances of the newer algorithms could be identified as either better coping with angular momentum (*L*) or, contraction (*K*); each new method increasing the speed and application of quantum mechanics to solving real chemical problems.

By 1990, another barrier had been reached. The contemporary programs had become sophisticated and both academia and industry had begun to recognize and use the power of *ab initio* quantum chemistry, but the software was struggling with "dusty deck syndrome" and it had become increasingly difficult for it to keep up with the rapid advances in hardware development. Vector processors, parallel architectures and the advent of the graphical user interface were all demanding radically different approaches to programming and it had become clear that a fresh start, with a clean slate, was both inevitable and desirable. Furthermore, the integral bottleneck had re-emerged in a new guise and the standard programs were now hitting the N^2 wall. Irrespective of the speed at which ERIs could be computed, the unforgiving fact remained that the number of ERIs required scaled quadratically with the size of the system.

The Q-Chem project was established to tackle this problem and to seek new methods that circumvent the N^2 wall. Fundamentally new approaches to integral theory were sought and the ongoing advances that have resulted [14-18] have now placed Q-Chem firmly at the vanguard of the field. It should be emphasized, however, that the $O(N)$ methods that we have developed still require short-range ERIs to treat interactions between nearby electrons, thus the importance of contemporary ERI code remains.

The chronological development and evolution of integral methods can be summarised by considering a timeline showing the years in which important new algorithms were first introduced. These are best discussed in terms of the type of ERI or matrix elements that the algorithm can compute efficiently.

1950	Boys	[11]	ERIs with low L and low K
1969	Pople	[13]	ERIs with low L and high K
1976	Dupuis	[19]	Integrals with any L and low K
1978	McMurchie	[20]	Integrals with any L and low K
1982	Almlöf	[21]	Introduction of the direct SCF approach
1986	Obara	[22]	Integrals with any L and low K
1988	Head-Gordon	[8]	Integrals with any L and low K
1991	Gill	[1, 6]	Integrals with any L and any K
1994	White	[14]	J matrix in linear work
1996	Schwegler	[18, 24]	HF exchange matrix in linear work
1997	Challacombe	[17]	Fock matrix in linear work

B.3 AOINTS: Calculating ERIs with Q-Chem

The area of molecular integrals with respect to Gaussian basis functions has recently been reviewed [2] and the user is referred to this review for deeper discussions and further references to the general area. The purpose of this short account is to present the basic approach, and in particular, the implementation of ERI algorithms and aspects of interest to the user in the AOINTS package which underlies the Q-Chem program.

We begin by observing that all of the integrals encountered in an *ab initio* calculation, of which overlap, kinetic energy, multipole moment, internuclear repulsion, nuclear-electron attraction and interelectron repulsion are the best known, can be written in the general form

$$\left(\mathbf{ab}|\mathbf{cd}\right) = \iint \phi_{\mathbf{a}}(\mathbf{r_1})\phi_{\mathbf{b}}(\mathbf{r_1})\theta(r_{12})\phi_{\mathbf{c}}(\mathbf{r_2})\phi_{\mathbf{d}}(\mathbf{r_2})\,\mathrm{d}\,\mathbf{r_1}\,\mathrm{d}\,\mathbf{r_2} \qquad \text{(B.1)}$$

where the basis functions are contracted Gaussian’s (CGTF)

$$\phi_{\mathbf{a}}(\mathbf{r}) = \left(x - A_x\right)^{a_x}\left(y - A_y\right)^{a_y}\left(z - A_z\right)^{a_z}\sum_{i=1}^{K_a} D_{ai}e^{-\alpha_i|\mathbf{r}-\mathbf{A}|^2} \qquad \text{(B.2)}$$

and the operator θ is a two-electron operator. Of the two-electron operators (Coulomb, CASE, anti-Coulomb and delta-function) used in the Q-Chem program, the most significant is the Coulomb, which leads us to the ERIs.

An ERI is the classical Coulomb interaction ($\theta(x) = 1/x$ in B.1) between two charge distributions referred to as bras (**ab**| and kets |**cd**).

B.3.1 Shell-Pair Data

It is common to characterise a bra, a ket and a bra-ket by their degree of contraction and angular momentum. In general, it is more convenient to compile data for shell-pairs rather than basis-function-pairs. A shell is defined as that sharing common exponents and centres. For example, in the case of a number of Pople derived basis sets, four basis functions, encompassing a range of angular momentum types (*i.e.*, *s*, p_x, p_y, p_z) on the same atomic centre sharing the same exponents constitute a single shell.

The shell-pair data set is central to the success of any modern integral program for three main reasons. First, in the formation of shell-pairs, all pairs of shells in the basis set are considered and categorized as either significant or negligible. A shell-pair is considered negligible if the shells involved are so far apart, relative to their diffuseness, that their overlap is negligible. Given the rate of decay of Gaussian basis functions, it is not surprising that most of the shell-pairs in a large molecule are negligible, that is, the number of significant shell-pairs increases linearly with the size of the molecule. Second, a number of useful intermediates which are frequently required within ERI algorithms should be computed once in shell-pair formation and stored as part of the shell-pair information, particularly those which require costly divisions. This prevents re-evaluating simple quantities. Third, it is useful to sort the shell-pair information by type (*i.e.*, angular momentum and degree of contraction). The reasons for this are discussed below.

Q-Chem's shell-pair formation offers the option of two basic integral shell-pair cutoff criteria; one based on the integral threshold (*$rem* variable *THRESH*) and the other relative to machine precision.

Intelligent construction of shell-pair data scales linearly with the size of the basis set, requires a relative amount of CPU time which is almost entirely negligible for large direct SCF calculations, and for small jobs, constitutes approximately 10% of the job time.

B.3.2 Shell-Quartets and Integral Classes

Given a sorted list of shell-pair data, it is possible to construct all potentially important shell-quartets by pairing of the shell-pairs with one another. Because the shell-pairs have been sorted, it is possible to deal with batches of integrals of the same type or class (*e.g.*, (*ss*|*ss*), (*sp*|*sp*), (*dd*|*dd*), *etc*.) where an integral class is characterized by both angular momentum (*L*) and degree of contraction (*K*). Such an approach is advantageous for vector processors and for semi-direct integral algorithms where the most expensive (high *K* or *L*) integral classes can be computed once, stored in memory (or disk) and only less expensive classes rebuilt on each iteration.

While the shell-pairs may have been carefully screened, it is possible for a pair of significant shell-pairs to form a shell-quartet which need not be computed directly. Three cases are:

1. The quartet is equivalent, by point group symmetry, to another quartet already treated.
2. The quartet can be ignored on the basis of cheaply computed ERI bounds [7] on the largest quartet bra-ket.
3. On the basis of an incremental Fock matrix build, the largest density matrix element which will multiply any of the bra-kets associated with the quartet may be negligibly small.

Note: Significance and negligibility is always based on the level of integral threshold set by the *$rem* variable *THRESH*.

B.3.3 FUNDAMENTAL ERI

The fundamental ERI [2] and the basis of all ERI algorithms is usually represented

$$\begin{aligned}[\mathbf{0}]^{(0)} &= [ss|ss]^{(0)} \\ &= D_A D_B D_C D_D \iint e^{-\alpha|\mathbf{r_1}-\mathbf{A}|^2} e^{-\beta|\mathbf{r_1}-\mathbf{B}|^2} \left[\frac{1}{r_{12}}\right] e^{-\gamma|\mathbf{r_2}-\mathbf{C}|^2} e^{-\delta|\mathbf{r_2}-\mathbf{D}|^2}\,\mathrm{d}\,\mathbf{r_1}\,\mathrm{d}\,\mathbf{r_2}\end{aligned} \tag{B.3}$$

which can be reduced to a one-dimensional integral of the form

$$[\mathbf{0}]^{(0)} = U(2\vartheta^2)^{1/2}\left(\frac{2}{\pi}\right)^{1/2}\int_0^1 e^{-Tu^2}\,\mathrm{d}\,u \tag{B.4}$$

and can be efficiently computed using a modified Chebyshev interpolation scheme [5]. Equation (B.4) can also be adapted for the general-case $[\mathbf{0}]^{(m)}$ integrals required for most calculations. Following the fundamental ERI, building up to the full bra-ket ERI (or intermediary matrix elements, see later) are the problems of angular momentum and contraction.

Note: Square brackets denote primitive integrals and parentheses fully contracted.

B.3.4 ANGULAR MOMENTUM PROBLEM

The fundamental integral is essentially an integral without angular momentum (*i.e.*, it is an integral of the type [*ss*|*ss*]). Angular momentum, usually depicted by *L*, has been problematic for efficient ERI formation, evident in the timeline in section B.2. Initially, angular momentum was calculated by taking derivatives of the fundamental ERI with respect to one of the Cartesian coordinates of the nuclear centre. This is an extremely inefficient route, but it works and was appropriate in the early development of ERI methods. Recursion relations [22, 25] and the newly developed tensor equations [3] are the basis for the modern approaches.

B.3.5 CONTRACTION PROBLEM

The contraction problem may be described by considering a general contracted ERI of *s*-type functions derived from the STO-3G basis set. Each basis function has degree of contraction $K = 3$. Thus, the ERI may be written

$$
\begin{aligned}
(ss|ss) &= \sum_{i=1}^{3}\sum_{j=1}^{3}\sum_{k=1}^{3}\sum_{l=1}^{3} D_{Ai}D_{Bj}D_{Ck}D_{Dl} \times \\
&\iint e^{-\alpha_i|\mathbf{r_1}-\mathbf{A}|^2} e^{-\beta_j|\mathbf{r_1}-\mathbf{B}|^2} \left[\frac{1}{r_{12}}\right] e^{-\gamma_k|\mathbf{r_2}-\mathbf{C}|^2} e^{-\delta_l|\mathbf{r_2}-\mathbf{D}|^2}\, \mathrm{d}\mathbf{r_1}\,\mathrm{d}\mathbf{r_2} \qquad \text{(B.5)} \\
&= \sum_{i=1}^{3}\sum_{j=1}^{3}\sum_{k=1}^{3}\sum_{l=1}^{3} [s_i s_j|s_k s_l]
\end{aligned}
$$

and requires 81 primitive integrals for the single ERI. The problem escalates dramatically for more highly contracted sets (STO-6G, 6-311G) and has been the motivation for the development of techniques for shell-pair modelling [26] in which a second shell-pair is constructed with fewer primitives that the first, but introduces no extra error relative to the integral threshold sought.

The Pople-Hehre axis-switch method [13] is excellent for high contraction low angular momentum integral classes.

B.3.6 QUADRATIC SCALING

The success of quantitative modern quantum chemistry, relative to its primitive, qualitative beginnings, can be traced to two sources: better algorithms and better computers. While the two technologies continue to improve rapidly, efforts are heavily thwarted by the fact that the total number of ERIs increases quadratically with the size of the molecular system. Even large increases in ERI algorithm efficiency yield only moderate increases in applicability, hindering the more widespread application of *ab initio* methods to areas of, perhaps, biochemical significance where semi-empirical techniques [27, 28] have already proven so valuable.

Thus, the elimination of quadratic scaling algorithms has been the theme of many research efforts in quantum chemistry throughout the 1990's and has seen the construction of many alternative algorithms to alleviate the problem. Johnson was the first to implement DFT exchange/correlation functionals whose computational cost scaled linearly with system size [23]. This paved the way for the most significant breakthrough in the area with the linear scaling CFMM algorithm [14] leading to linear scaling DFT calculations [29]. Further breakthroughs have been made with traditional theory in the form of the QCTC [17, 30, 31]and ONX [18, 24] algorithms, whilst more radical approaches [15, 16, 32] may lead to entirely new approaches to *ab initio* calculations. Investigations into the quadratic Coulomb problem has not only yielded linear scaling algorithms, but is also providing large insights into the significance of many molecular energy components.

Linear scaling Coulomb and SCF exchange/correlation algorithms are not the end of the story as the $O(N^3)$ diagonalization step has been rate limiting in semi-empirical techniques and, been predicted [33] to become rate limiting in *ab initio* approaches in the medium term. However, divide-and-conquer techniques [34-37] and the recently developed quadratically convergent SCF algorithm [38] show great promise for reducing this problem.

B.3.7 Algorithm Selection

No single ERI algorithm is available to efficiently handle all integral classes; rather, each tends to have specific integral classes where the specific algorithm out-performs the alternatives. The PRISM algorithm [6] is an intricate collection of pathways and steps in which the path chosen is that which is the most efficient for a given class. It appears that the most appropriate path for a given integral class depends on the relative position of the contraction step (lowly contracted bra-kets prefer late contraction, highly contracted bra-kets are most efficient with early contraction steps).

Careful studies have provided FLOP counts which are the current basis of integral algorithm selection, although care must be taken to ensure that algorithms are not rate limited by MOPs [4]. Future algorithm selection criteria will take greater account of memory, disk, chip architecture, cache size, vectorization and parallelization characteristics of the hardware, many of which are already exist within Q-Chem.

B.3.8 User Controllable Variables

AOINTS has been optimally constructed so that the fastest integral algorithm for ERI calculation is chosen for the given integral class and batch. Thus, the user has not been provided with the necessary variables for over-riding the program's selection process. The user is, however, able to control the accuracy of the cutoff used during shell-pair formation (*METECO*) and the integral threshold (*THRESH*). In addition, the user can force the use of the direct SCF algorithm (*DIRECT_SCF*) and increase the default size of the integrals storage buffer (*INCORE_INTS_BUFFER*).

Currently, some of Q-Chem's linear scaling algorithms, such as QCTC and ONX algorithms, require the user to specify their use. It is anticipated that further research developments will lead to the identification of situations in which these, or combinations of these and other algorithms, will be selected automatically by Q-Chem in much the same way that PRISM algorithms choose the most efficient pathway for given integral classes.

B.4 REFERENCES

[1] P. M. W. Gill, M. Head-Gordon and J. A. Pople, *J. Phys. Chem.* (1990), **94**, 5564.

[2] P. M. W. Gill, *Advan. Quantum Chem.* (1994), **25**, 142.

[3] T. R. Adams, R. D. Adamson and P. M. W. Gill, *J. Chem. Phys.* (1997), **107**, 124.

[4] M. J. Frisch, B. G. Johnson, P. M. W. Gill, D. J. Fox and R. H. Nobes, *Chem. Phys. Lett.* (1993), **206**, 225.

[5] P. M. W. Gill, B. G. Johnson and J. A. Pople, *Int. J. Quantum Chem.* (1991), **40**, 745.

[6] P. M. W. Gill and J. A. Pople, *Int. J. Quantum Chem.* (1991), **40**, 753.

[7] P. M. W. Gill, B. G. Johnson and J. A. Pople, *Chem. Phys. Lett.* (1994), **217**, 65.

[8] M. Head-Gordon and J. A. Pople, *J. Chem. Phys.* (1988), **89**, 5777.

[9] B. G. Johnson, P. M. W. Gill and J. A. Pople, *Chem. Phys. Lett.* (1993), **206**, 229.

[10] B. G. Johnson, P. M. W. Gill and J. A. Pople, *Chem. Phys. Lett.* (1993), **206**, 239.

[11] S. F. Boys, *Proc. Roy. Soc., A* (1950), **A200**, 542.

[12] S. F. Boys, G. B. Cook, C. M. Reeves and I. Shavitt, *Nature* (1956), **178**, 1207.

[13] J. A. Pople and W. J. Hehre, *J. Comput. Phys.* (1978), **27**, 161.

[14] C. A. White, B. G. Johnson, P. M. W. Gill and M. Head-Gordon, *Chem. Phys. Lett.* (1994), **230**, 8.

[15] R. D. Adamson, J. P. Dombroski and P. M. W. Gill, *Chem. Phys. Lett.* (1996), **254**, 329.

[16] J. P. Dombroski, S. W. Taylor and P. M. W. Gill, *J. Phys. Chem.* (1996), **100**, 6272.

[17] M. Challacombe and E. Schwegler, *J. Chem. Phys.* (1997), **106**, 5526.

[18] E. Schwegler and M. Challacombe, *J. Chem. Phys.* (1996), **105**, 2726.

[19] M. Dupuis, J. Rys and H. F. King, *J. Chem. Phys.* (1976), **65**, 111.

[20] L. E. McMurchie and E. R. Davidson, *J. Comput. Phys.* (1978), **26**, 218.

[21] J. Almlöf, K. Faegri and K. Korsell, *J. Comput. Chem.* (1982), **3**, 385.

[22] S. Obara and A. Saika, *J. Chem. Phys.* (1986), **84**, 3963.

[23] B. G. Johnson, PhD Thesis. "Development, implementation and performance of efficient methodologies for density functional calculations" Carnegie Mellon: Pittsburgh, 1993.

[24] E. Schwegler and M. Challacombe, *J. Chem. Phys.* (1996), **106**, 9708.

[25] S. Obara and A. Saika, *J. Chem. Phys.* (1988), **89**, 1540.

[26] R. D. Adamson, Honours Thesis. "Shell-pair economisation" Massey University: Palmerston North, 1995.

[27] M. J. S. Dewar, *Org. Mass. Spec.* (1993), **28**, 305.

[28] M. J. S. Dewar, *The molecular orbital theory of organic chemistry*; McGraw-Hill: New York, 1969.

[29] C. A. White, B. G. Johnson, P. M. W. Gill and M. Head-Gordon, *Chem. Phys. Lett.* (1996), **253**, 268.

[30] M. Challacombe, E. Schwegler and J. Almlöf, *J. Chem. Phys.* (1996), **104**, 4685.

[31] M. Challacombe, E. Schwegler and J. Almlöf, "Modern developments in Hartree-Fock theory: Fast methods for computing the Coulomb matrix," Department of Chemistry and Minnesota Supercomputer Institute, University of Minnesota, 1995.

[32] A. M. Lee, R. D. Adamson and P. M. W. Gill, *to be published*.

[33] D. L. Strout and G. E. Scuseria, *J. Chem. Phys.* (1995), **102**, 8448.

[34] W. Yang, *Phys. Rev. Lett.* (1991), **66**, 1438.

[35] W. Yang, *Phys. Rev. A* (1991), **44**, 7823.

[36] W. Yang and T.-S. Lee, *J. Chem. Phys.* (1995), **103**, 5674.

[37] T.-S. Lee, D. M. York and W. Yang, *J. Chem. Phys.* (1996), **105**, 2744.

[38] C. Ochsenfeld and M. Head-Gordon, *Chem. Phys. Lett.* (1997), **270**, 399.

APPENDIX C $REM VARIABLE REFERENCE

C.1 INPUT DECK FORMAT

The general format of the *$rem* input for Q-Chem text input files is simply as follows:

```
$rem
rem_variable   rem_option     [comment]
rem_variable   rem_option     [comment]
$end
```

The following sections contain the names and options of available *$rem* variables for users. The format for describing each *$rem* variable is as follows

REM_VARIABLE

Gives a short description of what the variable controls
VARIABLE:
Defines the variable as either INTEGER, LOGICAL or STRING
DEFAULT:
Describes Q-Chem's internal default, if any exist
OPTIONS:
Lists options available for the user
RECOMMENDATION:
Gives a quick recommendation

C.2 GENERAL

BASIS

Defines the basis sets to be used
VARIABLE:
STRING
DEFAULT:
No default basis set
OPTIONS:

General, Gen	User defined. (*$basis* keyword required)
Symbol	Use standard basis sets as per Chapter 6

RECOMMENDATIONS:
Consult literature and reviews to aid your selection

JOBTYPE

Specifies the calculation

VARIABLE:

STRING

DEFAULT:

SP	Single point energy

OPTIONS:

SP	Single point energy
FORCE	Analytical Force calculation
OPT	Geometry Minimization
TS	Transition Structure Search
FREQ	Frequency Calculation

RECOMMENDATION:

Defaults to single point

MEMORY

Sets the memory for individual program modules

VARIABLE:

INTEGER

DEFAULT:

2,000,000 (2 MW)

OPTIONS:

User-defined number of words

MEMORY_TOTAL

Sets the total memory available to Q-Chem

VARIABLE:

INTEGER

DEFAULT:

Unlimited (1,000 MW)

OPTIONS:

User-defined number of words

RECOMMENDATION:

Use default

PURECART

Controls the use of either pure or Cartesian basis functions for general basis sets (ignored for standard basis sets)

DEFAULT:

None

OPTIONS:

gfd	Use 1 for pure and a 2 for Cartesian for each angular momentum type. Must be defined for user supplied basis sets

SYMMETRY

Controls the use of efficiency through the use of point group symmetry

VARIABLE:

LOGICAL

DEFAULT:

TRUE	Use symmetry when available

OPTIONS:

TRUE	Use symmetry when available
FALSE	Do not use symmetry

RECOMMENDATION:

Use default

UNRESTRICTED

Controls the use of restricted or unrestricted orbitals

VARIABLE:

LOGICAL

DEFAULT:

FALSE	(Restricted) Closed-shell systems
TRUE	(Unrestricted) Open-shell systems

OPTIONS:

True	Unrestricted orbitals
False	Restricted open-shell HF (ROHF)

RECOMMENDATION:

Use default unless ROHF is desired

C.3 SCF GROUND STATE CALCULATIONS

See also *BASIS* and *JOBTYPE*

CORRELATION

Specifies the correlation level of theory

VARIABLE:

STRING

DEFAULT:

None	No Correlation

OPTIONS:

None	No Correlation
VWN	Vosko-Wilk-Nusair parameterization #5
LYP	Lee-Yang-Parr (LYP)
PW91, PW	GGA91 (Perdew)
LYP(EDF1)	LYP(EDF1) parameterization
Perdew86, P86	Perdew 1986
PZ81, PZ	Perdew-Zunger 1981
Wigner	Wigner
MP2	MP2
CIS(D)	CIS(D) (excited states)

DIIS_SUBSPACE_SIZE

Controls the size of the DIIS subspace during the SCF

VARIABLE:

INTEGER

DEFAULT:

15

OPTIONS:

User-defined

DIRECT_SCF

Controls direct SCF

VARIABLE:

LOGICAL

DEFAULT:

Determined by program

OPTIONS:

TRUE	Forces direct SCF
FALSE	Do not use direct SCF

RECOMMENDATION:

Use default. *DIRECT_SCF* switches off in-core integrals

EXCHANGE

Specifies the exchange level of theory

VARIABLE:

STRING

DEFAULT:

No default

OPTIONS:

HF	Exact (Hartree-Fock)
Slater, S	Slater
Becke, B	Becke
Gill96, Gill	Gill 1996
Becke(EDF1), B(EDF1)	Becke (EDF1)
PW91, PW	Perdew
B3PW91, Becke3PW91, B3P	B3PW91 hybrid
B3LYP, Becke3LYP	B3LYP hybrid
EDF1	EDF1
General, Gen	User defined combination of K, X and C (refer DFT section, Chapter 4)

INCORE_INTS_BUFFER

Controls the size of in-core integral storage buffer

VARIABLE:

INTEGER

DEFAULT:

2,000,000 words (1 word = 8 bytes).

OPTIONS:

User defined size; hardware dependent

INTEGRAL_2E_OPR

Determines the two-electron operator

VARIABLE:

INTEGER

DEFAULT:

-2	Coulomb Operator

OPTIONS:

-1	Apply the CASE approximation
-2	Coulomb Operator

MAX_SCF_CYCLES

Controls the maximum number of SCF iterations permitted

VARIABLE:

INTEGER

DEFAULT:

50

OPTIONS:

User-defined

METECO

Sets the threshold criteria for discarding shell-pairs

VARIABLE:

INTEGER

DEFAULT:

2 Discard shell-pairs below $10^{-\mathrm{THRESH}}$.

OPTIONS:

1 Discard shell-pairs fours orders of magnitude below machine precision

2 Discard shell-pairs below $10^{-\mathrm{THRESH}}$

RECOMMENDATION:

Use default

OMEGA

Controls the degree of attenuation of the Coulomb operator

VARIABLE:

INTEGER

DEFAULT:

No default

OPTIONS:

n $\omega = n/1000$

SCF_ALGORITHM

Algorithm used for converging the SCF

VARIABLE:

STRING

DEFAULT:

DIIS Pulay DIIS

OPTIONS:

DIIS Pulay DIIS

DM Direct minimizer

DIIS_DM Uses DIIS initially, switching to direct minimizer for later iterations

ROOTHAAN Roothaan repeated diagonalization

RECOMMENDATION:

Use DIIS unless wanting ROHF, in which case use the direct minimiser must be used

SCF_CONVERGENCE

SCF is considered converged when the wavefunction error is less that $10^{-\mathrm{SCF\_CONVERGENCE}}$

VARIABLE:

INTEGER

DEFAULT:

5 For single point energy calculations

8 for geometry optimizations and vibrational analysis

OPTIONS:

User-defined

RECOMMENDATION:

Tighter criteria for geometry optimization and vibration analysis

SCF_GUESS

Specifies the initial guess procedure to use for the SCF

VARIABLE:

STRING

DEFAULT:

SAD Superposition of atomic density (available only with standard basis sets)

OPTIONS:

CORE Diagonalize core Hamiltonian

SAD Superposition of atomic density

GWH Apply generalized Wolfsberg-Helmholtz approximation

READ Read previous MOs from disk

RECOMMENDATION:

SAD guess for standard basis sets. For general basis sets, use GWH guess and switch to core Hamiltonian guess if problems are encountered

SCF_GUESS_ALWAYS

Switch to force the regeneration of a new initial guess for each series of SCF iterations (for use in geometry optimization)

VARIABLE:

LOGICAL

DEFAULT:

False Do not generate a new guess for each series of SCF iterations in an optimization; use MOs from the previous SCF calculation for the guess, if available

OPTIONS:

False Do not generate a new guess for each series of SCF iterations in an optimization; use MOs from the previous SCF calculation for the guess, if available

True Generate a new guess for each series of SCF iterations in a geometry optimization

SCF_GUESS_MIX

Switch to control mixing of LUMO and HOMO to break symmetry in the initial guess

VARIABLE:

LOGICAL

DEFAULT:

False	Do not mix HOMO and LUMO in SCF guess

OPTIONS:

False	Do not mix HOMO and LUMO in SCF guess
True	Add 10% of LUMO to HOMO for SCF initial guess to break symmetry

THRESH

Cutoff for neglect of two electron integrals. $10^{-\text{THRESH}}$ (*THRESH* ≤ 12)

VARIABLE:

INTEGER

DEFAULT:

8	for single point energies
10	for optimizations and frequency calculations

OPTIONS:

User-defined

RECOMMENDATION:

Should be at least three greater than *SCF_CONVERGENCE*

XC_GRID

Specifies the type of grid to use for DFT calculations.

DEFAULT:

1	SG-1

OPTIONS:

1	SG-1
2	Low Quality
mn	The first six integers correspond to *m* radial points and the second six integers correspond to *n* angular points where possible numbers of Lebedev angular points are listed in Chapter 4
-*mn*	The first six integers correspond to *m* radial points and the second six integers correspond to *n* angular points where the number of Gauss-Legendre angular points $n = 2N_\theta^2$

RECOMMENDATION:

SG-1

C.4 LARGE MOLECULES

See also *JOBTYPE*, *BASIS*, *EXCHANGE* and *CORRELATION*

CFMM_ORDER

Controls the order of multipole expansions in CFMM calculation

VARIABLE:

INTEGER

DEFAULT:

15	For single point SCF accuracy
25	For tighter convergence

OPTIONS:

n	Use multipole expansions of order *n*

RECOMMENDATION:

Use default

GRAIN

Controls the number of lowest-level boxes in one-dimension for CFMM

VARIABLE:

INTEGER

DEFAULT:

-1	Program decides best value, turning CFMM on when useful

OPTIONS:

-1	Program decides best value, turning CFMM on when useful
1	Do not use CFMM
n>7	Use CFMM with *n* lowest-level boxes in one-dimension

RECOMMENDATION:

This is an expert option; either use the default, or use a value of 1 if CFMM is not desired

INCFOCK

Iteration number after which the incremental Fock matrix algorithm is initiated

VARIABLE:

INTEGER

DEFAULT:

1	Start INCFOCK after iteration number 1

OPTIONS:

User-defined (0 switches INCFOCK off)

RECOMMENDATIONS:

May be necessary to allow several iterations before switching on INCFOCK

ONX

Switch for ONX linear scaling algorithm for Hartree-Fock Exchange.

VARIABLE:

LOGICAL

DEFAULT:

False	Do not use ONX

OPTIONS:

True	Use ONX (requires CFMM or QCTC)
False	Do not use ONX

ONX_MXDIS

Memory allocation control variable for ONX

VARIABLE:

INTEGER

DEFAULT:

50,000 (words)

OPTIONS:

User-defined

RECOMMENDATIONS:

May require increasing for large jobs

ONX_MXPRM

Memory allocation control variable for ONX

VARIABLE:

INTEGER

DEFAULT:

100,000 (words)

OPTIONS:

User-defined

RECOMMENDATIONS:

May require increasing for large jobs

QCTC

Switch for the Quantum Chemical Tree Code linear scaling algorithm for Coulomb interactions

VARIABLE:

LOGICAL

DEFAULT:

False	Do not use QCTC

OPTIONS:

True	Use QCTC
False	Do not use QCTC

QCTC_MAXBIGMEM

Memory allocation control variable for QCTC

VARIABLE:

INTEGER

DEFAULT:

300,000 (words)

OPTIONS:

User-defined

RECOMMENDATIONS:

May need increasing for large jobs.

Note: may be necessary especially for crashes in *maketree* and *initrho* subroutines.

VARTHRESH

Controls the temporary integral cutoff threshold.

tmp_thresh = $10^{-\text{VARTHRESH}}$ x DIIS_error

VARIABLE:

INTEGER

DEFAULT:

3

OPTIONS:

User-defined threshold

RECOMMENDATIONS:

3 has found to be a practical level

C.5 CORRELATED METHODS

See also *JOBTYPE*, *BASIS*, *EXCHANGE* and *CORRELATION*

CD_ALGORITHM

Determines the algorithm for integral transformations

VARIABLE:

STRING

DEFAULT:

Program determined

OPTIONS:

DIRECT	Uses fully direct algorithm
SEMI_DIRECT	Uses disk-based sortless semi-direct algorithm

RECOMMENDATION:

Use the default

CD_MAX_DISK

Sets the amount of disk space (in words) available for integral transforms

VARIABLE:

INTEGER

DEFAULT:

60,000,000 (60 MW)

OPTIONS:

User-defined

N_FROZEN_CORE

Controls the number of frozen core orbitals in a post-Hartree-Fock calculation

VARIABLE:

INTEGER

DEFAULT:

0	No frozen core orbitals

OPTIONS:

FC	Frozen core approximation (all core orbitals frozen)
n	Freeze *n* core orbitals

N_FROZEN_VIRTUAL

Controls the number of frozen virtual orbitals in a post-Hartree-Fock calculation

VARIABLE:

INTEGER

DEFAULT:

0	No frozen virtual orbitals

OPTIONS:

n	Freeze *n* virtual orbitals

C.6 EXCITED STATES

See also *JOBTYPE*, *BASIS*, *EXCHANGE* and *CORRELATION*

CD_ALGORITHM

Determines the algorithm for integral transformations

VARIABLE:

STRING

DEFAULT:

Program-determined

OPTIONS:

DIRECT	Uses fully direct algorithm
SEMI_DIRECT	Uses disk-based sortless semi-direct algorithm

RECOMMENDATION:

Use the default

CD_MAX_DISK

Sets the amount of disk space (in words) available for integral transforms

VARIABLE:

INTEGER

DEFAULT:

60,000,000 (60 MW)

OPTIONS:

User-defined

CIS_CONVERGENCE

CIS is considered converged when error is less than $10^{-\text{CIS\_CONVERGENCE}}$

VARIABLE:

INTEGER

DEFAULT:

6	CIS convergence threshold ≡ 10^{-6}

OPTIONS:

User-defined

CIS_GUESS_DISK

Read the CIS guess from disk (previous calculation).

VARIABLE:

LOGICAL

DEFAULT:

False	Create a new guess

OPTIONS:

False	Create a new guess
True	Read the guess from disk

RECOMMENDATIONS:

Requires a guess from previous calculation

CIS_GUESS_DISK_TYPE

Determines the type of guesses to be read from disk

VARIABLE:

INTEGER

DEFAULT:

None

OPTIONS:

0	Read triplets only
1	Read triplets and singlets
2	Read singlets only

RECOMMENDATIONS:

Must be specified if *CIS_GUESS_DISK* is *TRUE*

CIS_N_ROOTS

Sets the number of CI-Singles (CIS) excited state roots to find

VARIABLE:

INTEGER

DEFAULT:

0 Do not look for any excited states

OPTIONS:

n > 0 Looks for *n* CIS excited states

CIS_RELAXED_DENSITY

Use the relaxed CIS density for attachment/detachment density analysis

VARIABLE:

LOGICAL

DEFAULT:

False Do not use the relaxed CIS density in analysis

OPTIONS:

False	Do not use the relaxed CIS density in analysis
True	Use the relaxed CIS density in analysis

CIS_SINGLETS

Solve for singlet excited states in RCIS calculations (ignored for UCIS)

VARIABLE:

LOGICAL

DEFAULT:

True Solve for singlet states

OPTIONS:

True	Solve for singlet states
False	Do not solve for singlet states

CIS_STATE_DERIV

Sets which state to determine CIS gradient for excited state optimizations

VARIABLE:

INTEGER

DEFAULT:

0	Does not select any of the excited states

OPTIONS:

n	Select the n^{th} state

CIS_TRIPLETS

Solve for triplet excited states in RCIS calculations (ignored for UCIS)

VARIABLE:

LOGICAL

DEFAULT:

True	Solve for triplet states

OPTIONS:

True	Solve for triplet states
False	Do not solve for triplet states

MAX_CIS_CYCLES

Maximum number of CIS iterative cycles allowed

VARIABLE:

INTEGER

DEFAULT:

30

OPTIONS:

User-defined

RECOMMENDATIONS:

Default is usually sufficient

N_FROZEN_CORE

Controls the number of frozen core orbitals

VARIABLE:

INTEGER

DEFAULT:

0	No frozen core orbitals

OPTIONS:

FC	Frozen core approximation
n	Freeze *n* core orbitals

N_FROZEN_VIRTUAL

Controls the number of frozen virtual orbitals
VARIABLE:
INTEGER
DEFAULT:

0	No frozen virtual orbitals

OPTIONS:

n	Freeze *n* virtual orbitals

RPA

Do an RPA calculation in addition to a CIS calculation
VARIABLE:
LOGICAL
DEFAULT:

False	Do not do an RPA calculation

OPTIONS:

False	Do not do an RPA calculation
True	Do an RPA calculation

XCIS

Do an XCIS calculation in addition to a CIS calculation
VARIABLE:
LOGICAL
DEFAULT:

False	Do not do an XCIS calculation

OPTIONS:

False	Do not do an XCIS calculation
True	Do an XCIS calculation (requires ROHF ground state)

C.7 Molecular Geometry Critical Points

See also *JOBTYPE*, *BASIS*, *EXCHANGE* and *CORRELATION*

GEOM_OPT_COORDS

Controls the type of optimization coordinates

VARIABLE

INTEGER

DEFAULT:

-1 Generate and optimize in internal coordinates, if this fails at any stage of the optimization, switch to Cartesian and continue

OPTIONS:

0	Optimize in Cartesian coordinates
1	Generate and optimize in internal coordinates, if this fails abort
-1	Generate and optimize in internal coordinates, if this fails at any stage of the optimization, switch to Cartesian and continue
2	Optimize in *Z*-matrix coordinates, if this fails abort
-2	Optimize in *Z*-matrix coordinates, if this fails during any stage of the optimization switch to Cartesians and continue

RECOMMENDATION:

Use the default; delocalized internals are more efficient

GEOM_OPT_DMAX

Maximum allowed step size. Value supplied is multiplied by 10^{-3}

VARIABLE:

INTEGER

DEFAULT:

300 = 0.3

OPTIONS:

n User-defined cutoff

RECOMMENDATION:

Use default

GEOM_OPT_HESSIAN

Hessian status

VARIABLE:

INTEGER

DEFAULT:

DIAGONAL Set up (default) diagonal Hessian

OPTIONS:

DIAGONAL	Set up (default) diagonal Hessian
READ	Have exact or initial Hessian. Use as is if Cartesian or transform if internals

GEOM_OPT_LINEAR_ANGLE

Threshold for near linear bond angles (degrees)

VARIABLE:

INTEGER

DEFAULT:

165 degrees

OPTIONS:

n	User-defined level

RECOMMENDATION:

Use default

GEOM_OPT_MAX_CYCLES

Maximum number of optimisation cycles

VARIABLE:

INTEGER

DEFAULT:

20

OPTIONS:

User defined positive integer

RECOMMENDATION:

Use the default, increase for difficult cases

GEOM_OPT_MAX_DIIS

Controls maximum size of subspace for GDIIS

VARIABLE:

INTEGER

DEFAULT:

0	Do not use GDIIS

OPTIONS:

0	Do not use GDIIS
-1	Default size = min(NDEG, NATOMS, 4) NDEG = number of molecular degrees of freedom
n	Size specified by user

RECOMMENDATION:

Use default or do not set *n* too large

GEOM_OPT_MODE

Determines Hessian mode followed during TS search
VARIABLE:
INTEGER
DEFAULT:

0	Mode following off

OPTIONS:

0	Mode following off
n	Maximise along mode *n*

RECOMMENDATION:
Use default

GEOM_OPT_TOL_DISPLACEMENT

Convergence on maximum atomic displacement
VARIABLE:
INTEGER
DEFAULT:

1200	≡ 1200 x 10^{-6} tolerance on maximum atomic displacement

OPTIONS:
Integer value (tolerance = value x 10^{-6})
RECOMMENDATION:
Use the default. To converge *GEOM_OPT_TOL_GRADIENT* and one of *GEOM_OPT_TOL_DISPLACEMENT* and *GEOM_OPT_TOL_ENERGY* must be satisfied

GEOM_OPT_TOL_ENERGY

Convergence on energy change of successive optimisation cycles
VARIABLE:
INTEGER
DEFAULT:

100	≡ 100 x 10^{-8} tolerance on maximum gradient component

OPTIONS:
Integer value (tolerance = value x 10^{-8})
RECOMMENDATION:
Use the default. To converge *GEOM_OPT_TOL_GRADIENT* and one of *GEOM_OPT_TOL_DISPLACEMENT* and *GEOM_OPT_TOL_ENERGY* must be satisfied

GEOM_OPT_TOL_GRADIENT

Convergence on maximum gradient component

VARIABLE:

INTEGER

DEFAULT:

300 ≡ 300 x 10^{-6} tolerance on maximum gradient component

OPTIONS:

Integer value (tolerance = value x 10^{-6})

RECOMMENDATION:

Use the default. To converge *GEOM_OPT_TOL_GRADIENT* and one of *GEOM_OPT_TOL_DISPLACEMENT* and *GEOM_OPT_TOL_ENERGY* must be satisfied

GEOM_OPT_SYMFLAG

Controls the use of symmetry in OPTIMIZE

VARIABLE:

INTEGER

DEFAULT:

1 Make use of point group symmetry

OPTIONS:

1	Make use of point group symmetry
0	Do not make use of point group symmetry

RECOMMENDATION:

Use default

GEOM_OPT_UPDATE

Controls the Hessian update algorithm

VARIABLE:

INTEGER

DEFAULT:

-1 Use the default update algorithm

OPTIONS:

-1	Use the default update algorithm
0	Do not update the Hessian (not recommended)
1	Murtagh-Sargent update
2	Powell update
3	Powell-Murtagh-Sargent update (TS default)
4	BFGS update (OPT default)
5	BFGS with safeguards to ensure retention of positive definiteness (GDISS default)

RECOMMENDATION:

Use default

C.8 MOLECULAR PROPERTIES AND ANALYSIS

DORAMAN

Controls calculation of Raman intensities. Requires *JOBTYPE* to be set to *FREQ*

VARIABLE:

LOGICAL

DEFAULT:

FALSE Do not calculate Raman intensities

OPTIONS:

FALSE Do not calculate Raman intensities

TRUE Do calculate Raman intensities

MULTIPOLE_ORDER

Determines highest order to of multipole moments to print if wave function analysis requested

VARIABLE:

INTEGER

DEFAULT:

4

OPTIONS:

n Calculate moments to n^{th} order

NBO

Controls the use of the NBO package

VARIABLE:

LOGICAL

DEFAULT:

FALSE Do not invoke the NBO package

OPTIONS:

FALSE Do not invoke the NBO package

TRUE Do invoke the NBO package

POP_MULLIKEN

Controls running of Mulliken population analysis

VARIABLE:

LOGICAL

DEFAULT:

TRUE Calculate Mulliken population

OPTIONS:

TRUE Calculate Mulliken population

FALSE Do not calculate Mulliken Population

RECOMMENDATIONS:

TRUE. Trivial additional calculation

SOLUTE_RADIUS

Sets the Onsager solvent model cavity radius

VARIABLE:

INTEGER a_0 = SOLUTE_RADIUS/10000

DEFAULT:

No default

OPTIONS:

User-defined

RECOMMENDATION:

Use equation (8.1)

SOLVENT_DIELECTRIC

Sets the dielectric constant of the Onsager solvent continuum

VARIABLE:

INTEGER ε = SOLVENT_DIELECTRIC/10000

DEFAULT:

No default

OPTIONS:

User-defined

RECOMMENDATION:

As per required solvent

SYMMETRY_DECOMPOSITION

Determines symmetry decompositions to calculate

VARIABLE:

INTEGER

DEFAULT:

1 Calculate MO eigenvalues and symmetry (if available)

OPTIONS:

0	No symmetry decomposition
1	Calculate MO eigenvalues and symmetry (if available)
2	Perform symmetry decomposition of kinetic energy and nuclear attraction matrices

WAVEFUNCTION_ANALYSIS

Controls the running of the default wave function analysis

VARIABLE:

LOGICAL

DEFAULT:

TRUE Perform default wave function analysis

OPTIONS:

TRUE	Perform default wave function analysis.
FALSE	Do not perform default wave function analysis

C.9 PRINTING

DIIS_PRINT

Controls the output from DIIS SCF optimization

VARIABLE:

INTEGER

DEFAULT:

0

OPTIONS:

0

1 Chosen method and DIIS coefficients & solutions

2 Level 1 plus changes in multipole moments

3 Level 2 plus Multipole moments

4 Level 3 plus extrapolated Fock matrices

GEOM_OPT_PRINT

Amount of OPTIMIZE print output

VARIABLE:

INTEGER

DEFAULT:

3 Error messages, summary, warning, standard information and gradient print out

OPTIONS:

0 Error messages only

1 Level 0 plus summary and warning print out

2 Level 1 plus standard information

3 Level 2 plus gradient print out

4 Level 3 plus hessian print out

5 Level 4 plus iterative print out

6 Level 5 plus internal generation print out

7 Debug print out

RECOMMENDATION:

Use the default

PRINT_GENERAL_BASIS

Controls print out of built in basis sets in input format

VARIABLE:

LOGICAL

DEFAULT:

False Do not print out standard basis set information

OPTIONS:

TRUE Print out standard basis set information

FALSE Do not print out standard basis set information

RECOMMENDATIONS:

Useful for modification of standard basis sets

SCF_FINAL_PRINT

Controls level of output from SCF procedure to Q-Chem output file at the end of the SCF

VARIABLE:

INTEGER

DEFAULT:

0 No extra print out

OPTIONS:

0 No extra print out
1 Orbital Energies only
2 Level 1 plus MOs
3 Level 2 plus Fock and density matrices

SCF_GUESS_PRINT

Controls printing of guess MOs, Fock and density matrices

VARIABLE:

INTEGER

DEFAULT:

0 Do not print guesses

OPTIONS:

0 Do not print guesses

SAD

1 Atomic density matrics and molecular matrix
2 Level 1 plus density matrices

CORE and GWH

1 No extra output
2 Level 1 plus Fock and density matrices and, MO coefficients and eigenvalues

READ

1 No extra output
2 Level 1 plus density matrices, MO coefficients and eigenvalues

SCF_PRINT

Controls level of output from SCF procedure to Q-Chem output file

VARIABLE:

INTEGER

DEFAULT:

0 Minimal, concise, useful and necessary output

OPTIONS:

0	Minimal, concise, useful and necessary output
1	Level 0 plus component breakdown of SCF electronic energy
2	Level 1 plus density, Fock an MO matrices on each cycle
3	Level 2 plus two-electron Fock matrix components (Coulomb, HF exchange and DFT exchange-correlation matrices) on each cycle

RECOMMENDATION:

Proceed with care; can result in *extremely* large output files at level 2 or higher

VIBMAN_PRINT

Controls level of extra print out for vibrational analysis

VARIABLE:

INTEGER

DEFAULT:

1 Standard full information print out

OPTIONS:

1	Standard full information print out
3	Level 1 plus vibrational frequencies in atomic units
4	Level 3 plus mass weighted Hessian matrix, projected mass-weighted Hessian matrix
6	Level 4 plus vectors for translations and rotations, projection matrix

RECOMMENDATION:

Use default

APPENDIX D SAMPLE INPUTS

Provided in the following section are sample inputs which you may copy and use. The index below lists the sample input found in this guide and supplied on the distribution media.

Filename	**Molecule**	**Comment**	**Procedure**
B3LYP_water.in	water	Hybrid functional	B3LYP/6-31G*
BASIS_water.in	water	User-defined basis	HF/General
BATCH.in	hydrogen	Batch job	OPT HF/6-31G* FREQ HF/6-31G* SP MP2/6-311G(d,p)
CASE.in	water	CASE FREQ	HF/6-31g(df,p)
CFMM_azt.in	azt	Use CFMM	HFS/3-21G
CIS.in	water	CIS single point	RCIS/6-31G*
CIS_disk.in	water	CIS guess from disk	RCIS/STO-3G (singlets) RCIS/STO-3G (singlets)
DFT_azt.in	azt	Large molecule	HFS/3-21G
DFT_benzene.in	benzene	DFT, ints, symm	EDF1/6-31+G*
DFT_cyclohex.in	cyclohexane	DFT single point	B-LYP/6-31G*
DFT_cycloprop.in	cyclopropane	DFT single point	Gill-P86/TZ*
DFT_ferrocene.in	ferrocene	GWH guess	S-VWN/VDZ*
DFT_glutamine.in	glutamine	GGA-DFT	PW91-PW91/6-311G
DFT_tryptophan.in	tryptophan	Large molecule	S-PZ/DZ
FREQ_fluoro.in	fluoromethane	OPT & FREQ	HF/6-31G
FREQ_water.in	water	Vib frequencies	HF/6-31G*
HESS_OPT.in	water	Begin opt with analytic Hessian	HF/6-31G(d)
GOGC_methanol.in	methanol	Opt. constraints	HF/3-21G
HF_asp.in	asparagine	HF Single point	HF/STO-3G
HF_methanol.in	methanol	Hartree-Fock	HF/6-31+G*
HF_water.in	water	Hartree-Fock (tight)	HF/3-21G*
MP2_fluoro.in	fluoromethane	MP2 single point	MP2/6-311G(d,p)
OPT_methanol.in	methanol	Optimization	S-VWN/3-21G
OPT_water.in	water	Optimization	B-LYP/6-31G*
RO_excited.in	methyl	XCIS/RPA Frozen core	XCIS/STO-3G + RPA XCIS/3-21G + RPA
U_R_CIS_RPA.in	methyl water	UCIS/URPA RCIS/RRPA	UCIS/STO-3G + RPA RCIS/STO-3G + RRPA

Filename	Molecule	Comment	Procedure
B3LYP_water.in	water	Hybrid functional	B3LYP/6-31G*

```
$comment
water  B3LYP/6-31G*  Single point energy
$end

$molecule
0 1
O
H1 O OH
H2 O OH H1 HOH

OH  = 0.947
HOH = 105.5
$end

$rem
EXCHANGE           B3LYP
CORRELATION        None    Hybrid functional, correlation incorporated
BASIS              6-31G*  Basis Set
$end
```

Filename	**Molecule**	**Comment**	**Procedure**
BASIS_water.in	water	User-defined basis	HF/General

```
$molecule
0 1
O
H O OH
H O OH 2 HOH

OH = 0.957
HOH = 104.5
$end

$rem
EXCHANGE        HF        HF Exchange
CORRELATION     None      No Correlation
BASIS           General   User-defined general basis
SCF_GUESS       GWH
$end

$basis
H       0
S       2        1.00
        1.30976  0.430129
        0.233136 0.678914
****
O       0
S       2        1.00
       49.9810   0.430129
        8.89659  0.678914
SP      2        1.00
        1.94524  0.0494720  0.511541
        0.493363 0.963782   0.612820
****
$end

@@@
$molecule
0 1
O
H O OH
H O OH 2 HOH

OH = 0.957
HOH = 104.5
$end

$rem
EXCHANGE        HF        HF Exchange
CORRELATION     None      No correlation
BASIS           General   User-defined general basis
SCF_GUESS       GWH
PURECART        1         Pure D functions
$end

$basis
H       0
S       2        1.00
```

```
        1.30976  0.430129
        0.233136 0.678914
****
O       0
S       2        1.00
       49.9810   0.430129
        8.89659  0.678914
SP      2        1.00
        1.94524  0.0494720  0.511541
        0.493363 0.963782   0.612820
D       1        1.00
        0.39     1.00000
****
$end

@@@
$molecule
0 1
O
H O OH
H O OH 2 HOH

OH = 0.957
HOH = 104.5
$end

$rem
EXCHANGE        HF        HF Exchange
CORRELATION     None      No correlation
BASIS           General   User-defined general basis
SCF_GUESS       GWH
PURECART        221       Pure D, Cartesian F and G
$end

$basis
H       0
S       2        1.00
        1.30976  0.430129
        0.233136 0.678914
****
O       0
S       2        1.00
        49.9810  0.430129
        8.89659  0.678914
SP      2        1.00
        1.94524  0.0494720   0.511541
        0.493363 0.963782    0.612820
D       1        1.00
        0.39     1.00000
F       1        1.00
        4.1      1.00
G       1       1.00
        3.35    1.00
****
$end

@@@
$molecule
0 1
O
```

```
H O OH
H O OH 2 HOH

OH = 0.957
HOH = 104.5
$end

$rem
EXCHANGE        HF        HF Exchange
CORRELATION     None      No correlation
BASIS           General   User-defined general basis
SCF_GUESS       GWH
$end

$basis
H       0
STO-2G
****
O       0
STO-6G
****
$end

@@@
$molecule
0 1
O
H O OH
H O OH 2 HOH

OH = 0.957
HOH = 104.5
$end

$rem
EXCHANGE        HF        HF Exchange
CORRELATION     None      No correlation
BASIS           General   User-defined general basis
SCF_GUESS       GWH
$end

$basis
H       0
S       2         1.00
        1.30976   0.430129
        0.233136  0.678914
****
O       0
STO-6G
****
$end

@@@
$molecule
0 1
O
H O OH
H O OH 2 HOH

OH = 0.957
```

```
HOH = 104.5
$end

$rem
EXCHANGE        HF        HF Exchange
CORRELATION     None      No Correlation
BASIS           General   User-defined general basis
SCF_GUESS       GWH
PURECART        2         Cartesian D functions
$end

$basis
H       0
6-31G
****
O       0
6-311G(d)
****
$end
```

Filename	**Molecule**	**Comment**	**Procedure**
BATCH.in	hydrogen	Batch job	OPT HF/6-31G* FREQ HF/6-31G* SP MP2/6-311G(d,p)

```
$molecule
0 1
H
H 1 r

r = 1.1
$end
$comment
Optimize H-H at HF/6-31G*
$end
$rem
JOBTYPE         OPT       Optimize the bond length
EXCHANGE        HF
BASIS           6-31G*
$end

Let's do another job while we are here ...

@@@
Use the optimized geometry from above
$molecule
READ
$end

$comment
Frequencies of H-H at HF/6-31G*
$end

Use the old MO's as this will save on the SCF

$rem
JOBTYPE         FREQ    Calculate vibrational frequencies
EXCHANGE        HF
BASIS           6-31G*
SCF_GUESS       READ    Read the MO's from disk
$end

Get a better single point energy ...

@@@
$comment
H-H at MP2/6-311G(d,p)//HF/6-31G*
$end
$molecule
READ
$end
$rem
EXCHANGE        HF
CORRELATION     MP2
BASIS           6-311G(d,p)
$end
```

Filename	**Molecule**	**Comment**	**Procedure**
CASE.in	water	CASE FREQ	HF/6-31g(df,p)

```
$comment
CASE test
$end

$molecule
0 1
o
h1 o 0.96857
h2 o 0.96857 h1 103.9807
$end

$rem
JOBTYP   FREQ
EXCHANGE HF
BASIS 6-31g(df,p)
OMEGA 200
INTEGRAL_2E_OPR -1
NBO   OFF
THRESH     12
KONSCF     10
$end
```

Filename	**Molecule**	**Comment**	**Procedure**
CFMM_azt.in	azt	Use CFMM	HFS/3-21G

```
$comment
azt    HFS/3-21G   Single point energy
$end

$molecule
0  1
O       0.866884     0.684479     1.438526
C       1.826996     1.127800     0.483147
C       2.384657    -0.116007    -0.209336
N       3.775093    -0.310687     0.205595
N       4.596954    -0.622662    -0.657013
N       5.458532    -0.903949    -1.370990
C       1.490617    -1.258547     0.284435
C       0.421408    -0.592035     1.138572
N      -0.901017    -0.517967     0.456172
C      -1.486632    -1.743061     0.203515
O      -0.899859    -2.805202     0.413593
N      -2.767430    -1.663809    -0.306719
C      -3.522973    -0.510397    -0.507168
O      -4.666919    -0.592387    -0.960510
C      -2.857099     0.725793    -0.134700
C      -1.587458     0.662869     0.345991
C      -3.626454     1.988853    -0.294145
O       0.648063     3.148589     0.170040
C       1.244727     2.080871    -0.526701
H       2.080494     2.414263    -1.185118
H       0.507030     1.545393    -1.171345
H       0.412309     3.850305    -0.477858
H      -1.029301     1.556948     0.672661
H      -3.074099     2.851029     0.119957
H      -3.845322     2.181645    -1.360291
H      -4.605637     1.914247     0.211641
H      -3.219503    -2.561515    -0.512812
H       1.048368    -1.871220    -0.523936
H       2.085100    -1.934259     0.916312
H       0.244025    -1.141000     2.077353
H       2.619260     1.647513     1.056875
H       2.330267    -0.000482    -1.307602
$end

$rem
EXCHANGE                S         Slater exchange
BASIS                   3-21G     Basis Set
GRAIN                   8         Split 1-D space
CFMM_MULTIPOLE_ORDER    15
$end
```

Filename	Molecule	Comment	Procedure
CIS.in	water	CIS single point	RCIS/6-31G*

```
$comment
Water RCIS/6-31G*
$end

$molecule
0 1
O
H1 O OH
H2 O OH H1 HOH

OH  = 0.990
HOH = 100.0
$end

$rem
EXCHANGE      HF        HF Exchange
BASIS         6-31G*    Basis
CIS_N_ROOTS   3         Number of CIS states
CIS_SINGLETS  TRUE      Do Singlets
CIS_TRIPLETS  FALSE     Don't do Triplets
$end
```

Filename	**Molecule**	**Comment**	**Procedure**
CIS_disk.in	water	CIS guess from disk	RCIS/STO-3G (singlets) RCIS/STO-3G (singlets)

```
$comment
Water singlet RCIS/STO-3G
$end

$molecule
0 1
O
H1 O OH
H2 O OH H1 HOH

OH  = 0.990
HOH = 100.0
$end

$rem
EXCHANGE        HF       HF Exchange
CORRELATION     None     No Correlation
BASIS           STO-3G   basis
CIS_N_ROOTS     5        CIS roots
CIS_SINGLETS    TRUE     Do singlets
CIS_TRIPLETS    FALSE    Don't do triplets
$end

@@@
$comment
Water singlet/triplets RCIS/STO-3G (restart: singlets on disk)
$end

$molecule
0 1
O
H1 O OH
H2 O OH H1 HOH

OH  = 0.990
HOH = 100.0
$end

$rem
EXCHANGE             HF        HF Exchange
BASIS                STO-3G    basis
SCF_GUESS            READ      Read-in Guess
CIS_N_ROOTS          5         Number of CIS states
CIS_TRIPLETS         TRUE      Do triplets
CIS_SINGLETS         TRUE      Do singlets
CIS_GUESS_DISK       TRUE      Use CIS on disk as guess
CIS_GUESS_DISK_TYPE  2         Singlets read from disk
$end
```

Filename	**Molecule**	**Comment**	**Procedure**
DFT_azt.in	azt	Large molecule	HFS/3-21G

```
$comment
azt    HFS/3-21G   Single point energy
$end

$molecule
0  1
O       0.866884     0.684479     1.438526
C       1.826996     1.127800     0.483147
C       2.384657    -0.116007    -0.209336
N       3.775093    -0.310687     0.205595
N       4.596954    -0.622662    -0.657013
N       5.458532    -0.903949    -1.370990
C       1.490617    -1.258547     0.284435
C       0.421408    -0.592035     1.138572
N      -0.901017    -0.517967     0.456172
C      -1.486632    -1.743061     0.203515
O      -0.899859    -2.805202     0.413593
N      -2.767430    -1.663809    -0.306719
C      -3.522973    -0.510397    -0.507168
O      -4.666919    -0.592387    -0.960510
C      -2.857099     0.725793    -0.134700
C      -1.587458     0.662869     0.345991
C      -3.626454     1.988853    -0.294145
O       0.648063     3.148589     0.170040
C       1.244727     2.080871    -0.526701
H       2.080494     2.414263    -1.185118
H       0.507030     1.545393    -1.171345
H       0.412309     3.850305    -0.477858
H      -1.029301     1.556948     0.672661
H      -3.074099     2.851029     0.119957
H      -3.845322     2.181645    -1.360291
H      -4.605637     1.914247     0.211641
H      -3.219503    -2.561515    -0.512812
H       1.048368    -1.871220    -0.523936
H       2.085100    -1.934259     0.916312
H       0.244025    -1.141000     2.077353
H       2.619260     1.647513     1.056875
H       2.330267    -0.000482    -1.307602
$end

$rem
EXCHANGE           S         Slater exchange
CORRELATION        None      No correlation
BASIS              3-21G     Basis Set
$end
```

Filename	**Molecule**	**Comment**	**Procedure**
DFT_benzene.in	benzene	DFT, ints, symm	EDF1/6-31+G*

```
$comment
benzene EDF1/6-31+G* Single point energy
$end

$molecule
0  1
c1
c2 c1 cc
c3 c2 cc c1 120.0
c4 c3 cc c2 120.0 c1   0.0
c5 c4 cc c3 120.0 c2   0.0
c6 c5 cc c4 120.0 c3   0.0
h1 c1 hc c2 120.0 c3 180.0
h2 c2 hc c3 120.0 c4 180.0
h3 c3 hc c4 120.0 c5 180.0
h4 c4 hc c5 120.0 c6 180.0
h5 c5 hc c6 120.0 c1 180.0
h6 c6 hc c1 120.0 c2 180.0

cc = 1.3862
hc = 1.0756
$end

$rem
EXCHANGE              EDF1       Compound exchange-correlation
BASIS                 6-31+G*    Basis Set
INCORE_INTS_BUFFER    6000000
SYMMETRY              FALSE      Switch symmetry off
$end
```

Filename	**Molecule**	**Comment**	**Procedure**
DFT_cyclohex.in	cyclohexane	DFT single point	B-LYP/6-31G*

```
$comment
cyclohexane    B-LYP/6-31G*   Single point energy
$end

$molecule
0  1
6 -2.00217494  1.21765703  0.42917467
6 -0.48697492  1.21765703  0.42917467
1 -0.11683294  2.27652806  0.42917467
1 -0.11369297  0.72884002  1.36634768
6  0.07036701  0.48983705 -0.77654230
6 -0.48033091  1.06401903 -2.06493043
6 -1.99577794  1.04598004 -2.06888723
6 -2.55874094  1.77323705 -0.86651730
1 -2.30779594  2.86476702 -0.94051635
1 -3.67615494  1.68345103 -0.86340129
1 -2.35466495 -0.01707500 -2.06468129
1 -2.37451196  1.52595502 -3.00826144
1 -0.12142595  2.12041205 -2.18730735
1 -0.09476992  0.47497702 -2.93757128
1  1.18818203  0.56563401 -0.77676439
1 -0.19293090 -0.59871894 -0.70969438
1 -2.37648195  1.83084703  1.28912068
1 -2.37515893  0.16880602  0.57484267
$end

$rem
EXCHANGE           Becke     Becke88 exchange
CORRELATION        LYP       Lee-Yang-Parr correlation
BASIS              6-31G*    Basis Set
$end
```

Filename	**Molecule**	**Comment**	**Procedure**
DFT_cycloprop.in	cyclopropane	DFT single point	Gill-P86/TZ*

```
$comment
cyclopropane  Gill-P86/TZ*      Single point energy
$end

$molecule
0  1
6 -1.53561520 -0.55505953 -0.31674730
6 -0.03445816 -0.55505953 -0.31674730
6 -0.78504637  0.74489751 -0.31674730
1 -0.78501180  1.36429753 -1.23075168
1 -0.78497848  1.36431982  0.59723927
1  0.50213558 -0.86484615 -1.23063229
1  0.50211484 -0.86487678  0.59713991
1 -2.07205420 -0.86520455 -1.23059521
1 -2.07200771 -0.86518339  0.59714152
$end

$rem
EXCHANGE            Gill96   Gill96 exchange
CORRELATION         P86      Perdew 1986
BASIS               TZ*      Basis Set
SCF_GUESS           GWH      Generalized Wolfsberg-Helmholtz
$end
```

Filename	Molecule	Comment	Procedure
DFT_ferrocene.in	ferrocene	GWH guess	S-VWN/VDZ*

```
$comment
ferrocene    S-VWN/VDZ*   Single point energy
$end

$molecule
0 1
C       1.280000     1.310000     0.000000
C       1.280000     0.404800    -1.245900
C       1.280000    -1.059800    -0.770000
C       1.280000    -1.059800     0.770000
C       1.280000     0.404800     1.245900
C      -1.280000     1.059800    -0.770000
C      -1.280000    -0.404800    -1.245900
C      -1.280000    -1.310000     0.000000
C      -1.280000    -0.404800     1.245900
C      -1.280000     1.059800     0.770000
Fe      0.000000     0.000000     0.000000
H       1.280000     2.420000     0.000000
H       1.280000     0.747800    -2.301600
H       1.280000    -1.957800    -1.422400
H       1.280000    -1.957800     1.422400
H       1.280000     0.747800     2.301600
H      -1.280000     1.957800    -1.422400
H      -1.280000    -0.747800    -2.301600
H      -1.280000    -2.420000     0.000000
H      -1.280000    -0.747800     2.301600
H      -1.280000     1.957800     1.422400
$end

$rem
EXCHANGE           S      Slater exchange
CORRELATION        VWN    correlation
BASIS              VDZ*   Basis Set
SCF_GUESS          GWH    Generalized Wolfsberg-Helmholtz
$end
```

Filename	**Molecule**	**Comment**	**Procedure**
DFT_glutamine.in	glutamine	GGA-DFT	PW91-PW91/6-311G

```
$comment
glutamine    PW91-PW91/6-311G   Single point energy
$end

$molecule
0  1
6 -1.23973536  0.28356377  1.19837231
6 -2.02294832 -0.05092617 -0.05720877
6 -1.16425833  0.11331783 -1.30449280
6 -0.98613533  1.59214578 -1.68594879
1 -0.90978634  2.20609276 -0.75032580
7 -2.17109328  2.08999376 -2.42195075
1 -2.05935234  3.06007974 -2.62592882
1 -2.29374737  1.58214382 -3.27314382
6  0.30933064  1.74807278 -2.47708875
8  0.46060067  1.82271377 -3.68385774
8  1.43313271  1.82435183 -1.73186779
1  2.19459659  1.92147581 -2.29499584
1 -1.62354934 -0.45353122 -2.14053374
1 -0.17217135 -0.35508116 -1.14187479
1 -2.92050928  0.59796779 -0.12054282
1 -2.40883034 -1.08879312 -0.00201583
7 -0.37904635 -0.70950719  1.73604530
1 -0.69979234 -1.64213117  1.61126131
1 -0.04175234 -0.53298124  2.65596718
8 -1.24084032  1.38053079  1.73902721
$end

$rem
EXCHANGE           PW91     GGA91 exchange
CORRELATION        PW91     GGA91 correlation
BASIS              6-311G   Basis Set
$end
```

Filename	Molecule	Comment	Procedure
DFT_tryptophan.in	tryptophan	Large molecule	S-PZ/DZ

```
$comment
tryptophan    S-PZ/DZ   Single point energy
$end

$molecule
0  1
6 -1.76325628 -0.49073280 -0.55228146
6 -2.98203829 -1.30356275 -0.35232546
1 -3.85737828 -0.78281678 -0.79254346
1 -3.18504528 -1.37379876  0.73550455
6 -2.84471932 -2.72016670 -0.93460645
7 -2.61215332 -2.76292373 -2.39717742
1 -3.35090331 -2.29063578 -2.87543042
1 -1.74569729 -2.30701776 -2.59390839
6 -4.11333040 -3.50523879 -0.63144646
8 -5.25504521 -3.29227675 -0.99674647
8 -3.93999032 -4.58085574  0.16979853
1 -4.77101830 -5.02649583  0.30274252
1 -1.95588628 -3.22050478 -0.45377046
6 -0.41054526 -0.86925878 -0.21954145
6  0.15628871 -2.04650976  0.28680853
6  1.51987669 -2.08660378  0.48967354
6  2.33763167 -0.97639377  0.20013354
6  1.81060764  0.19604121 -0.30181545
6  0.42488667  0.24092720 -0.51077746
7 -0.40827433  1.29227816 -0.95520847
6 -1.72992230  0.80277930 -1.03612145
1 -2.56194833  1.41505013 -1.38761647
1 -0.07941833  2.07225715 -1.46414747
1  2.44560072  1.05749618 -0.52971546
1  3.41565367 -1.05384977  0.37688057
1  1.98547074 -2.99807776  0.88025157
1 -0.49007127 -2.91216780  0.49173657
$end

$rem
EXCHANGE           S     Slater exchange
CORRELATION        PZ    Perdew-Zunger correlation
BASIS              DZ    Basis Set
$end
```

Filename	**Molecule**	**Comment**	**Procedure**
FREQ_fluoro.in	fluoromethane	OPT & FREQ	HF/6-31G

```
$comment
Flurometnane   OPT   HF/6-31G
$end

$molecule
0  1
C
F  C RCF
H1 C RCH F HCF
H2 C RCH F HCF H1  120.0
H3 C RCH F HCF H1 -120.0

RCF = 1.383
RCH = 1.100
HCF = 108.3
$end

$rem
JOBTYPE            OPT           Optimization
EXCHANGE           HF            Exact exchange
BASIS              6-31G         Basis Set
$end

@@@
$comment
Flurometnane   FREQ   HF/6-31G
$end

Let's use data from the previous job ...
$molecule
READ
$end

$rem
JOBTYPE            FREQ          Vibrational frequencies
EXCHANGE           HF            Exact exchange
BASIS              6-31G         Basis Set
SCF_GUESS          READ
$end
```

Filename	**Molecule**	**Comment**	**Procedure**
FREQ_water.in	water	Vib frequencies	HF/6-31G*

```
$comment
water      HF/6-31G*   Vibrational frequencies
$end

$molecule
0  1
O
H1 O OH
H2 O OH H1 HOH

OH  = 0.947
HOH = 105.5
$end

$rem
JOBTYPE            FREQ     Vibrational frequencies
EXCHANGE           HF       HF exchange
BASIS              6-31G*   Basis Set
$end
```

Filename	Molecule	Comment	Procedure
HESS_OPT.in	water	Begin opt with analytic Hessian	HF/6-31G(d)

```
$molecule
0 1
O
H 1 OH
H 1 OH 2 HOH

OH = 1.1
HOH = 104
$end

$rem
JOBTYPE         FREQ
EXCHANGE        HF
BASIS           6-31G(D)
$end

Now proceed with the Optimization making sure to read in
the analytic Hessian ...

@@@
$molecule
READ
$end

$rem
JOBTYPE            OPT
EXCHANGE           HF
BASIS              6-31G(D)
SCF_GUESS          READ
GEOM_OPT_HESSIAN   READ
$end
```

Filename	**Molecule**	**Comment**	**Procedure**
GOGC_methanol.in	methanol	Opt. constraints	HF/3-21G

```
$comment
methanol geom opt with constraints in bond length and bond angles.
$end

$molecule
0  1
      c           0.141915    0.332682    0.000000
      o           0.141915   -1.088318    0.000000
      h           1.186989    0.656186    0.000000
      h          -0.348433    0.742676    0.887862
      h          -0.348433    0.742676   -0.887862
      h          -0.773953   -1.385902    0.000000
$end

$rem
GEOM_OPT_PRINT  6
GEOM_OPT_COORDS 2
nbo                off
JOBTYPE            opt
EXCHANGE           hf
BASIS              3-21g   Basis Set
$end

$opt
CONSTRAINT
stre 1 6  1.8
bend 2 1 4 110.0
bend 2 1 5 110.0
ENDCONSTRAINT
$end
```

Filename	**Molecule**	**Comment**	**Procedure**
HF_asp.in	asparagine	HF Single point	HF/STO-3G

```
$comment
asparagine    HF/STO-3G   Single point energy
$end

$molecule
0  1
6 -1.56579849  1.23838623  0.36392297
6 -1.92344249  1.08949823  1.84924896
1 -2.96945549  1.41311323  2.03541793
6 -0.98137350  1.87433120  2.74323298
7  0.38092159  1.47293922  2.79121699
1  0.52340960  0.49784323  2.65805401
1  0.91020655  1.84609422  3.54746999
8 -1.31274850  2.85359421  3.39258398
1 -1.88783849  0.01871624  2.13856902
1 -0.46414543  1.09804823  0.21861895
7 -1.87320248  2.60096422 -0.12156306
1 -1.69846349  2.65950325 -1.10157095
1 -2.82347752  2.83959725  0.06929197
6 -2.29777647  0.14641025 -0.41519402
8 -3.33787049  0.22502622 -1.04261694
8 -1.69859348 -1.06571087 -0.40673707
1 -2.21313150 -1.69037566 -0.90901099
$end

$rem
EXCHANGE           HF       Exact exchange
BASIS              STO-3G   Basis Set
$end
```

Filename	**Molecule**	**Comment**	**Procedure**
HF_methanol.in	methanol	Hartree-Fock	HF/6-31+G*

```
$comment
methanol    HF   6-31+G*   Single point energy
$end

$molecule
0  1
C
O  C RCO
H1 C RCH1 O H1CO
X  C 1.0  O XCO  H1  180.0
H2 C RCH2 X H2CX H1   90.0
H3 C RCH2 X H2CX H1  -90.0
H4 O ROH  C HOC  H1  180.0

RCO  = 1.421
RCH1 = 1.094
RCH2 = 1.094
ROH  = 0.963
H1CO = 107.2
XCO  = 129.9
H2CX = 54.25
HOC  = 108.0
$end

$rem
EXCHANGE           HF        Exact exchange
BASIS              6-31+G*   Basis Set
$end
```

Filename	**Molecule**	**Comment**	**Procedure**
HF_water.in	water	Hartree-Fock (tight)	HF/3-21G*

```
$comment
water    HF   3-21G*   Single point energy
$end

$molecule
0  1
O
H1 O OH
H2 O OH H1 HOH

OH  = 0.947
HOH = 105.5
$end

$rem
EXCHANGE           HF       Exact exchange
BASIS              3-21G*   Basis Set
SCF_CONVERGENCE    8        Tight convergence
$end
```

Filename	**Molecule**	**Comment**	**Procedure**
MP2_fluoro.in	fluoromethane	MP2 single point	MP2/6-311G(d,p)

```
$comment
Fluromethane   MP2/6-311G(d,p)   Single point energy
$end

$molecule
0  1
C
F  C RCF
H1 C RCH F HCF
H2 C RCH F HCF H1  120.0
H3 C RCH F HCF H1 -120.0

RCF = 1.383
RCH = 1.100
HCF = 108.3
$end

$rem
EXCHANGE           HF             Exact exchange
CORRELATION        MP2            MP2 correlation
BASIS              6-311G(d,p)    Basis Set
$end
```

Filename	**Molecule**	**Comment**	**Procedure**
OPT_methanol.in	methanol	Optimization	S-VWN/3-21G

```
$comment
methanol    S-VWN/3-21G      Geometry optimization
$end

$molecule
0  1
C
O  C RCO
H1 C RCH1 O H1CO
X  C 1.0  O XCO  H1  180.0
H2 C RCH2 X H2CX H1   90.0
H3 C RCH2 X H2CX H1  -90.0
H4 O ROH  C HOC  H1  180.0

RCO  = 1.421
RCH1 = 1.094
RCH2 = 1.094
ROH  = 0.963
H1CO = 107.2
XCO  = 129.9
H2CX = 54.25
HOC  = 108.0
$end

$rem
JOBTYPE          OPT      Geometry minimization
EXCHANGE         S        Slater exchange
CORRELATION      VWN      VWN correlation
BASIS            3-21G    Basis Set
$end
```

Filename	**Molecule**	**Comment**	**Procedure**
OPT_water.in	water	Optimization	B-LYP/6-31G*

```
$comment
water    B-LYP/6-31G*   Geometry Minimization
$end

$molecule
0 1
O
H 1 OH
H 1 OH 2 HOH

OH = 1.0
HOH = 100
$end

$rem
JOBTYPE           OPT     Geometry minimization
EXCHANGE          B       Becke exchange
CORRELATION       LYP     LYP correlation
BASIS             6-31G*  Basis Set
$end
```

Filename	Molecule	Comment	Procedure
RO_excited.in	methyl	XCIS/RPA	XCIS/STO-3G + RPA
		Frozen core	XCIS/3-21G + RPA

```
$comment
Methyl ROCIS & RORPA & XCIS & Quartet XCIS STO-3G
$end

$molecule
0,2
6   0.0000000000   0.0000000000   0.0000000000
1  -0.5346254069   0.9259983678   0.0000000000
1  -0.5346254069  -0.9259983678   0.0000000000
1   1.0692508139   0.0000000000   0.0000000000
$end

$rem
UNRESTRICTED    FALSE   Restricted orbitals
EXCHANGE        HF      HF exchange
BASIS           STO-3G
SCF_ALGORITHM   DM      Direct minimization
SCF_GUESS       GWH
CIS_N_ROOTS     5       Number of CIS/RPA/XCIS states
RPA             TRUE    Do RPA
XCIS            TRUE    Do XCIS
$end

@@@
$comment
Methyl ROCIS & quartet XCIS 3-21G Frozen Core
$end

$molecule
0,2
6   0.0000000000   0.0000000000   0.0000000000
1  -0.5346250000   0.9259976630   0.0000000000
1  -0.5346250000  -0.9259976630   0.0000000000
1   1.0692500000   0.0000000000   0.0000000000
$end

$rem
UNRESTRICTED  FALSE   Restricted orbitals
EXCHANGE      HF      HF Exchange
BASIS         3-21G
SCF_ALGORITHM DM      Direct minimization
SCF_GUESS     GWH
CIS_N_ROOTS   4       Number of CIS states
N_FROZEN_CORE 1       Number of frozen core orbitals
XCIS          TRUE    Do XCIS
$end
```

Filename	**Molecule**	**Comment**	**Procedure**
U_R_CIS_RPA.in	methyl	UCIS/URPA	UCIS/STO-3G + RPA
	water	RCIS/RRPA	RCIS/STO-3G + RRPA

```
$comment
Methyl UCIS & RPA STO-3G
$end

$molecule
0,2
 6   0.0000000000   0.0000000000   0.0000000000
 1  -0.5346254069   0.9259983678   0.0000000000
 1  -0.5346254069  -0.9259983678   0.0000000000
 1   1.0692508139   0.0000000000   0.0000000000
$end

$rem
EXCHANGE       HF       HF exchange
BASIS          STO-3G   Basis set
CIS_N_ROOTS    5        Number of CIS/RPA states
RPA            TRUE     Do an RPA as well as CIS
$end

@@@
$comment
Water singlet/triplet RCIS & RPA STO-3G
$end

$molecule
0,1
8  0.000  0.000000  0.128558
1  0.000  0.766044 -0.514230
1  0.000 -0.766044 -0.514230
$end

$rem
EXCHANGE      HF      HF Exchange
BASIS         STO-3G  STO-3G basis
CIS_N_ROOTS   5       Number of CIS/RPA states
RPA           TRUE    Do RPA
$end
```

INDEX

I.1 SUBJECT INDEX

I.2 KEYWORD INDEX